Issue 28 | Jun 2010

Blender learning made easy

blenderart MAGAZINE

Blender 2.5*

Creating 3d Contents With Blender 2.5

I love My Bubble

High Resolution Rendering at The Light of Speed

Blender 2.49 Sripting (Book Review)

COVERART Dikarya by Will Davis

EDITOR
Gaurav Nawani gaurav@blenderart.org

MANAGING EDITOR
Sandra Gilbert sandra@blenderart.org

WEBSITE
Nam Pham nam@blenderart.org

DESIGNER
Gaurav, Sandra, Alex

PROOFERS
Brian C. Treacy
Bruce Westfall
Daniel Hand
Daniel Mate
Henriël Veldtmann
Joshua Leung
Joshua Scotton
Kevin Braun
Mark Warren
Noah Summers
Patrick ODonnell
Phillip
Ronan Posnic
Scott Hill
Wade Bick
Valérie Hambert

WRITERS
by Francois “Coyhoyt” Grassard
Max Kielland
Nilson Juba & Jeff Israel
David Ward
Satish Goda
William Le Ferrand

COVER ART
Dikarya -by Will Davis

CONTENTS

Sandra Gilbert
Managing Editor

"Imagine my surprise when I opened it the first time and was promptly overcome with a 10 year old feeling of deja vu. "

I stumbled into the Blender universe about ten years ago, (for about 15 minutes)... at which time the massive number of buttons and options scared me so bad that I frantically deleted Blender from my hard drive and went to find something less scary to play with.

Obviously I found my way back to Blender (about a year later) and learned how to use it. But that first brush with Blender is something many of us are familiar with.

Fast forward ten odd years, Blender 2.5 series has been released. Imagine my surprise when I opened it the first time and was promptly overcome with a 10 year old feeling of deja vu. Oh snap! I'm a newbie again.

I'm sure I'm not the only one who received a momentary shock upon seeing the newest incarnation of Blender.

Luckily, the learning curve was much smoother this time. And while I occasionally need to poke around to find a familiar tool or feature, there is a beautiful logic and flow to Blender that makes the creative process so much easier these days.

The Blender 2.5 series has been out for a while now, and while it still is under going a lot of changes and refinement, it is stable enough for some serious testing and playing. So let's get everyone "Up to Speed" with Blender 2.5 and how to best take advantage of all the wonderful new toys and options available.

If you have not yet taken the 2.5 plunge, now is your chance to learn what the future holds ■

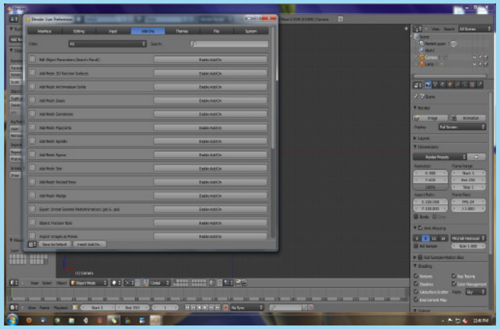
"My beloved spacebar now brings up a search menu instead of an add menu."

I have been using Blender for a long time and have developed my own little work flow that I have gotten rather used to.

With the release of the 2.5 series, my work flow has of course undergone numerous changes and adjustments. Most of them for the better. But one change, quite honestly, has continually tripped me up.

My beloved spacebar now brings up a search menu instead of an add menu. And yes, I have known for years that Shift + A, will do the same thing. But that isn't what I learned when I started. And of course, 10 years of muscle memory still finds me hitting the space bar and still being surprised when a search option comes up instead of an add menu.

Rather annoying, to say the least.



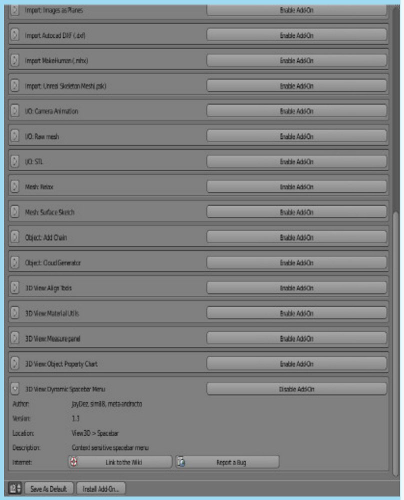
Having decided that I would just have to get used to it, I was overjoyed when I discovered that there was a wonderful new addition to Blender.

The "Add On" section of the User Preferences window. This lovely little window is populated with a number of "add on" extensions that can be enabled / disabled as you need. The addons set to enabled will of course load automatically when you launch Blender.

There are already a number of fun and useful add ons, but the one that makes my day is the "Dynamic Spacebar Menu".

When enabled it brings up a content sensitive menu full of useful options, including "Add Object".

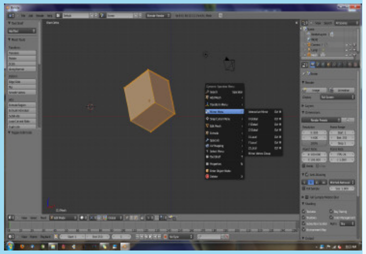
Yay me! My workflow is saved. And now I'm off to explore what else is hidden in the "Add On" section 🟥

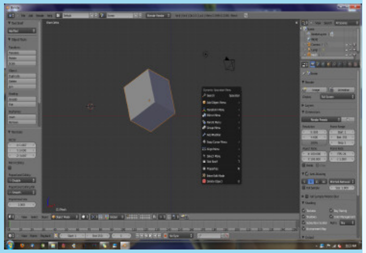


Import: Images as Planes	Enable Add-On
Import Autocad DXF (.dxf)	Enable Add-On
Import MakeHuman (.mhx)	Enable Add-On
Import: Unreal Skeleton Mesh (.psk)	Enable Add-On
I/O: Camera Animation	Enable Add-On
I/O: Raw mesh	Enable Add-On
I/O: STL	Enable Add-On
Mesh: Relax	Enable Add-On
Mesh: Surface Sketch	Enable Add-On
Object: Add Chain	Enable Add-On
Object: Cloud Generator	Enable Add-On
3D View: Align Tools	Enable Add-On
3D View: Material Utils	Enable Add-On
3D View: Measure panel	Enable Add-On
3D View: Object Property Chart	Enable Add-On
3D View: Dynamic Spacebar Menu	Disable Add-On

Author: JayDez, sim88, meta-androcto
Version: 1.3
Location: View3D > Spacebar
Description: Context sensitive spacebar menu
Internet: Link to the Wiki | Report a Bug

Save As Default | Install Add-On...





3D contents using Blender 2.5

by Francois "Coyhoyt" Grassard

Introduction

Since Avatar came up on the big screen, it's totally impossible to walk in the street and avoid hundreds of billboards showing a 3D logo. Over the last 3 months, 3D devices are everywhere ... and heavy marketing too ! « Are you still working on poor 2D images ? Dawn, you're so ridiculous, guy ! » That's the kind of sentences you can hear when you work for broadcasting since Avatar came out.

Producers and directors, all want to create 3D content. But in fact, what is the whole technology behind 3D images ? Is that really new ? « Of course ! » shouts together 20th Century Fox and Mister Cameron. « There is before Avatar and After Avatar ». What can be said after these kind of sentences ? The most humble answer we can provide is ... 1840. It deserves some explanations.

1840 is the date when the first 3D images were released. Yes, more than a century and a half before Avatar ! Surprising, isn't it ? In 1820, French guy Nicéphore Niépce created the first positive photograph. Only twenty years later, just after Niépce died, the first stereoscopic photography was done by another guy named Daguerre, but the whole process was known by scientists years before.

Old San Souci House, Old Orchard Beach, Maine, from Robert N. Dennis collection of stereoscopic views (~1870-1880)

Two images, one for each eye and slightly offset in space. Before photography, it's was really difficult for the painter to create exactly the same two paintings. When photography came up, it was easier to take two shots at the same time with two synchronized cameras. The stereoscopic view was born !

Really simple stereoscopic device

We will describe in detail the whole process in the next chapter, but if you are interested by the history of 3D images, I highly recommend the website http://photostereo.org created by a French guy, Francis Dupin. The website is bilingual (French and English) and contains a lot of stereoscopic photographs from the early age. Take a look at the « History » page. You will be probably surprised to discover that the 3D concept is quite old.

First, I'd like to clarify some things. I don't tell you that Avatar sucks. Technically, it's a really great movie. All the work done by different teams, like the wonderful forest shots from Weta are totally awesome.

No doubt about that. Modeling, rendering, lightning, mocap and facial animation, they're all great !

by Francois "Coyhoyt" Grassard

But I only complain about the marketing stuff around the movie, who tries to tell us that stereoscopy never existed before Avatar.

The goal of this article is to introduce the main concepts of stereoscopic images, also known as « 3D Images », the different parameters you have to take into account to produce good 3D and finally, how to make it with Blender 2.5 ! To do that, we will be accompanied by characters of Big buck Bunny, who will help us to understand all the concepts required.

A) How 3D works and how to produce it:

As we previously said, you need two 2D images to create one 3D image. When you look at an object in real life, your left eye and your right eye can see the same things but with a different point of view, just because they are not at the same place. With these two images, your brain creates a 3D representation of space, based essentially on parallax differences.

A.1) Parallax: The best friend of your brain:

Hey, « the best friend of your brain » ! That's a great advertising slogan, isn't it ? This magic word describes one of the most important concepts in 3D view. The one that is used by 3D tracking software to reconstruct a point cloud in 3D, extracted from only 2D images to finally create a 3D moving camera to match the real one.

To understand what it is, just make this simple experience with me. Put your index finger in front of your nose, about 10 cm away from it. Now, close your right eye and move your finger to place THIS WORD on the right of it.

Now, open your right eye and close the left one. The word has jumped to the other side of your finger ! That's because your finger is closer than the word. Each object, according the distance to your eyes, is horizontally offset when you switch from one eye to another. Far objects are minimally offset, close objects are highly offset. Parallax represents all the different offsets for your brain to create a mental 3D world.

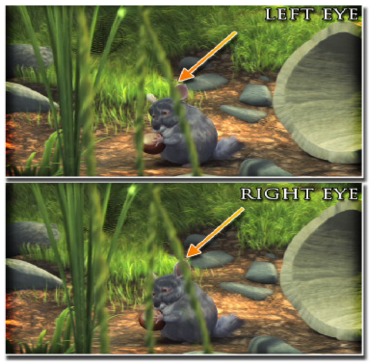
A.1) Parallax: The best friend of your brain:

In this case, chinchilla's ear is placed on the right of the grass for the left image, and on the left for the right image(First on next page). That's the parallax effect !

1

Two of the numerous 3D devices created by the past

By Moisés Espínola



A wonderful application of this concept, named Photogrammetry, can create a 3D model of a mountain from only two hi-res photos of it, shot by the same camera but slightly offset horizontally. High offset between the two same pixels represent close point, and low offset represent far point. A 3D point cloud can be extracted from the information and a mesh can be created from it, using Voronoi of Delaunay triangulation, for instance. Finally, one of the original images is projected on the mesh via camera mapping techniques, which provides a high-detailed and textured model of the mountain. Magic, isn't it ?

A.2) Interpupillary: This is not an insult !

The distance between the center of your eyes is called « Interpupillary » (IPD:InterPupillary Distance) and it's one of the keys of the stereoscopic view. The rotation of your eye doesn't change the IPD. Only the distance between the rotation centers is taken into account. So, you can keep a constant value even if you squint. This image of our friend Bunny, shows the IPD of his strange sight.

The average value of IPD for a human is 63mm (about 2.5 inches). Of course, a little boy of six years old doesn't have the same IPD as a basket player of 33 years old. The majority of adults have IPDs in the range 50–75 mm and the minimum IPD for a children is 40mm. We can considered that a baby of 3 months have a smaller IPD.

I hope you don't see that when you look into the mirror

But at this age, children don't care about any 3D images and prefer playing with mom's nipples. ;o)

So, that's the first thing to remember. The optical center of the two lenses of the two cameras, that they are real or virtual, has to be horizontally offset by the same value of this average IPD of 63mm and perfectly aligned vertically, as your eyes are. In Blender, the scale of your scene and all objects in it are thus important.

If you choose an IPD of 50, most of people will be disrupted all along the movie because this IPD will be too far from ours IPD. So, the average value of 63mm is the best choice because it is a medium value … for an adult. That's mean a 3D movie will be more difficult to « see » for a young child, because the difference between his IPD and the one used for the movie is higher than the one of his parents. It will require more effort for the eyes of a children to find the right vergence (we will explain this word in a few moments).

So, the choice of the IPD has to be made really carefully. If you choose an IPD of 95mm, that's means you work for an alien audience and for sure you'll give a human audience a big headache. I guess that's not your goal … except if you're an alien and have a plan to conquer the world.

by Francois "Coyboyt" Grassard

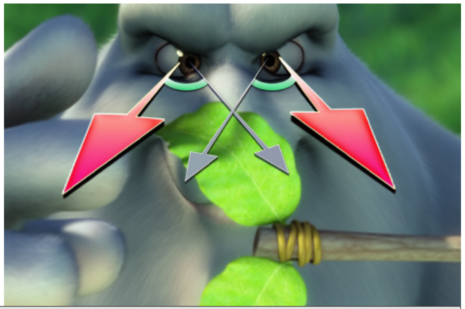
That's why the main 3D industry has to use this IPD of 63mm, choosen for parents who have enough money to pay for cinema ... children don't ! So, children can have an headache ... that's not a problem ... isn't it ? ;o)

A.3) 3D contents for Cinema or Television: Not the same fight:

As we just said, the IPD is the first parameter to take in account when you want to produce 3D contents. Close objects are highly offset between left eye and right eye, far objects are less offset. When you project a movie on the big screen, a little difference of parallax (how you produce the offset between left and right eye) can be enough to match your IPD.

Because the big screen is quite big. But if you watch the same movie on a small television, the relative distance between the two images is smaller because it is reduced. The « 3D effect » will be less impressive.

So, you have to think before your production is started, for what kind of medium your movie is made for and adapt the IPD used to the screen size. Theaters who want to display 3D movies have to have a minimum size for their screens. As an example, the post production of Avatar should be totally remade for small screens, if


To look at a close object, Bunny has to squint. Each eye rotates in opposite directions, according a vengeance angle

Mister Cameron wants to show the same experience to the audience who has seen his movie in theaters. Is that the reason why the release date of a BluRay version of Avatar in 3D is planned for the end of 2010 while the 2D version is already out ? The official explanation is that not enough people have a 3D BluRay player. That's probably a part of the truth, but my paranoid mind can't stop to trust in the other explanation. ;o)

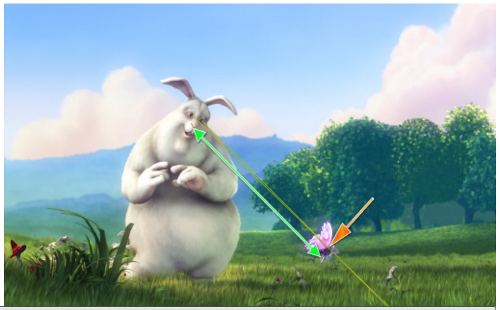
We will discuss this in a next chapter, 3D images can be recorded and broadcasted in many ways. But in each case, images for each eye can be extracted and process independently. A software like « Stereoscopic Player », can extract each image and reduce the offset between them just by a simple change of position, horizontally. I guess this feature will be available one day on BluRay 3D player and/or 3D television to virtually adapt the IPD to each size of screen and each viewer. But it's not enough to convert automatically a « Theaters Offset » to a « Television Offset », who probably required more work to achieve a good 3D TV experience.

A.4) Vergence and focal plane:

We previously described the concept of IPD. But there a second parameter that has the same importance. It's named « Vergence ». Once again, to understand this concept let's make together a second experience using your finger. Put your index finger in front of your nose, about 10 cm away from it and look at it. While you keep the focus on your finger, you can see behind it some parts of other objects, like a chair for instance. But you notice that you can see the chair twice. Now, if you keep the focus on the chair you can see your finger twice.

by Francois "Coyboyt" Grassard

When you keep the focus on your finger, each eye tries to rotate to clearly see the



When Bunny look at a far object, such as the butterfly, the vengeance angle is pretty low and lines extracted from eye are closed to be parallels

target, even if the rotation angle is different for each eye. You are simply squinting. Now, imagine two straight lines, one for each eye, starting from the center of the iris and going away in a direction corresponding to the rotation. In one point, those two lines intersect and create something called the « Focal Point »,formally the point where you look at. Stretched horizontally and vertically, this point can be extended to a normal plane named the « Focal Plane ».

When you watch a traditional (2D) TV, your eyes squint according to the distance from them to the screen. If your screen is far away from your eye, the convergence is low and the two lines are closed to be parallels. This situation is very relaxing for your eyes because muscles in charge of rotation are close to sleeping.

If the screen is too close to your eyes, the convergence has to be higher and lateral eye muscles work really hard, usually causing a headache.

In a 3D Workflow, the focal plane is used to place the position of your screen in the scene. All objects located between the camera and the focal plane can pop out of the screen. All objects located far away from the focal plane will look far away « behind » the screen. Objects located on the focal plane will be placed « in 3D space » exactly when your TV set is placed in your living room.

These parameters are probably the most important when you plan to create a movie of two hours. Imagine an action movie with two cuts per seconds, so ... a really speedy editing. For each shot, your eyes have to find where the focal plane is and adapt the vergence for it. If each shot are too short and the focal plane jumps from one position to another each second, it's headache day ! Because your eyes have to do a crazy gymnastics all along the movie.

When you switch from one shot to another, it's can really be uncomfortable. For instance, you have to shoot a Soccer match live. First cam shoot players from the top, really far from them and probably uses two cameras that are pretty close to be parallel. Suddenly, the ball has to be played as corner kick. We switch to a cam driven by a Steadicam.

placed only two meters behind the player who shoots the ball, with a high vergence angle. And Bam ! Your eyes have to converge to focus on the player who shoots the ball ... and Bam again, we switch back to the far cam. That's just a simple example, but it proves that we probably have to change the way a sport match is directed in 3D, to switch more smoothly from one cam to another. 90 minutes of eye gymnastics ... it's quite long ;o)

That's one of the secrets of Avatar. Why we don't have an headache after more than two hours of movie ? That's because all characters, and particularly their eyes, are always located on the focal plane. When you look at a character, you immediately look to their eyes. It's a reflex.

by Francois "Coyboyt" Grassard

By placing each character's eyes on focal plane, your eyes don't have to move at each cut when you switch from one character to another. In this case, a 3D movie is as comfortable as a 2D movie because your eyes converge always (or at least, most of the time) at the same point, where is the place of the screen. By this way, we avoid all that eye gymnastics. You can even watch the movie without any glasses ... all the characters won't have too much « blur » on their faces.

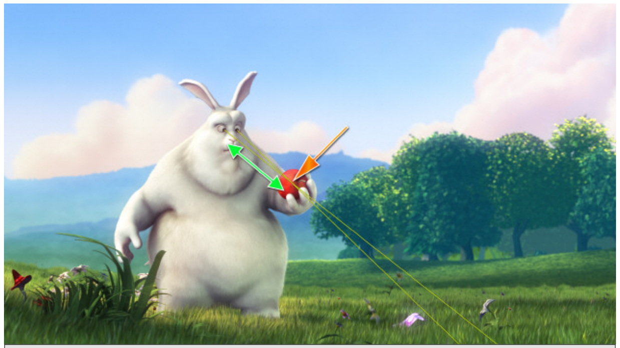
When you shoot a 3D movie, in real life or in CG world, you have to choose if your cameras will be totally parallel or use vergence. Both methods exist and are heavily discussed by 3D professionals. Parallel shooting is usually more comfortable because eye muscles don't have to work a lot. But with a « Parallel Rig », we consider that the focal plane is pushed to infinite. So, objects can pop up out of the screen, but no one can go far « behind it ».

When cameras use vergence, you can push objects far far away. But you have to adjust the rotation value of each cam really carefully. If it's too high, audience eyes will diverge. And your eyes never diverge in real life ! So, the final result of that is, once again, a big headache !

A.5) Optical issue in real life (why first 3D Movies was CGI):

We just discussed rotation of camera rigs using vergence. But what is exactly a 3D camera rig ?

In real life, a 3D camera rig is a set of tools that permit the placement of two cameras side by side and adjust the IPD and vergence between the cam. This kind of rig has to have a high degree of precision. All parameters changing on one cam have to be reported on the other cam. Focus, iris, zoom, gain, gamma .. and more. This synchronisation can be achieved by a mechanical or electronic process. Of course, the optical lens has to be the same for the two cameras. Many kinds of rigs exist. Just type « 3d camera rig » in Google image search engine to see dozen of different systems.


When Bunny look at a close object, such as the apple, the vengeance angle is high

Cameras are not always placed side by side. Because some kinds of camera are quite big ! Even if you placed the two cameras the closest you can, the distance between each optical center will be quite bigger than a human IPD. In this case, one cam can be placed like for a 2D shooting and the other one is placed upside down, filming the image reflected by a semi-transparent mirror.

Once again, type « 3d mirror rig » in Google image to see different systems used. There are many problems you have to manage when you shoot with this kind of rig. For instance, the camera that shoots the image that passes through the semi-transparent mirror is darker than the one is directly reflected and shot by the second cam (about 1 stop darker).

by Francois "Coyboyt" Grassard

So, you probably now understand that filming a 3D movie in real life is not so easy. Even if a special camera using two sensors and two lenses slowly came on the market, like the upcoming Panasonic camera or the robust one from « 3D One », stereoscopic shooting is a science and there are many things still to improve in this specific production pipeline.

When you create a 3D movie using only CGI tools, most of problems describe below disappear. Virtual cameras like the one you can handle in Blender don't have any size. So there is no problem about the IPD. Same thing about the rotation of camera according the vergence. In real life, the rotation angle of each camera has to be really precise. The value is usually around 2 or 3 degrees ! The rig controlling the two cameras have to be perfect, and obviously are costly. In Blender, setting the Y Rotation angle to a value of 0,25 degree is really easy. That is the main reason why most of 3D movies, for now at least, are CG.

A.6) Think about the limits … of your screen:

When you produce 3D contents, especially for 3DTV, you have to think about the limits of your screen, according to the field of view of your eyes. In a theater, if you're placed in front of the screen, pretty close to it, you don't pay any attention to the edge of the image. If a giant dinosaur jumps out of the screen (remember, closer than the focal plane), you have other things to do than looking the top right corner of the screen. Because, you're scared !!!

But when you watch a 3DTV, the edges of the screen are fully visible according to the field of view of your eyes. And it's pretty wide … around 160 degrees for most human (if you're a alien, let me know how your FOV is) ! There's a interesting experience to make if you have this kind of device. Put a simple cube between the focal plane and the camera rig. So, when you wear your 3D glasses, the cube seems to jump out of the screen. But if the cube comes closer, the edge of the screen will finally crop it. At this point, the cube seems to jump back into the screen very quickly, at a distance equal to the focal plane. Your eyes see some disparity between the two views by parallax differences. But your brain said that it's impossible to see an object at 1 meter away from your nose while the same object is cropped by the border of the screen, 2 meter behind.

So, if you can, you have to always keep objects that jump out of the screen inside the limits of that screen. If you can, that's another process that helps to limit the brain dilemma named « Floating Windows ».

The same problem appears when you shoot a panoramic shot, from left to right for instance … once again in a soccer match. Some elements of the image start to appear on the screen by the right view first, then in left view, one or more frames later. In this case, your two eyes don't see the same things on the edge of the TV picture. And that's bad for your brain ! So, the concept of floating Windows is quite simple. The goal is to hide in the right view all elements that you can't see into the left view. The problem is that you can't set the same crop value for all elements.

All objects have to be cropped according to their distance to the camera (remember the parallax and difference of speed between close and far objects). But this kind of « adaptive crop » is totally impossible in real life especially when you shoot live. So, we have to find a « generic solution » that's works for all images. The best solution is simply to slightly blur the side of the images. For the left view, you have to blur the left side a lot and right side a few, for right view … left side a few and right side a lot.

by Francois "Coyboyt" Grassard

Their blurry borders don't have to be wide, only 20 or 30 pixels on HD footage, and don't need a huge amount of blur. If the two cameras were perfectly aligned vertically during the shooting, horizontal blur is only needed.

This simple technique can strongly reduce this strange effect during a dolly or panoramic moving. I personally use it a lot when I work with virtual sets and Steadicam shots, with help of 3D tracking.

A.7) So many parameters to take into account:

As a conclusion to this big first part, we can say we described the most important parameters to take in account to produce 3D content. But in fact, more parameters should need more studies. For instance, the shading is one of the parameters that tells your brain some volumetric information. It's a big component of the 3D space representation created by your mind. So many things need to be analysed carefully. We are just at the beginning of the rise of the 3D wave … before it became a tsunami.

I hope this part was not too boring for you because it's not directly related to Blender. But before we describe some processes to create 3D contents using Blender, we have to describe what 3D is, right ?

Ok, now we know what 3D is and how it works, just take a look at how to broadcast it … and how we could broadcast it in the future.

B) Broadcasting 3D Contents:

When you want to broadcast 3D contents, you have to choose between two techniques :

- **First one** : Both images, for left and right eye, are projected at the same time, blended into one « composite » image. Glasses placed on your nose separate the two images, allowing each eye to see the right one. For this technique, we can use two kind of glasses, Anaglyph or Polarized, more generally called « Passive glasses ». We will described them further.
- **Second one** : Both images are projected sequentially, one after another. Left / Right / Left / Right / Left / Right / and so on. On your nose, you have to wear another kind of glasses, named « Active glasses ». They work by using powercell and are synchronized according to a reference signal emitted in the theater or by your 3DTV. When the projector shows an image for the left eye, glasses hide your right eye by activating a surface of LCD.

B.1) Three kind of glasses, three level of prices:

Ok, let's review the three kind of glasses :

Anaglyph glasses

Anaglyph : The goal of an Anaglyph image is to tint each « sub-image », for left and right eyes, with a different color and finally mix them into one image. There's not any strict standard defined for anaglyph images. But generally, the luminance of the left image is tinted using Red color at 100% while the luminance of the right image is tinted using Cyan color (composed of green and blue at 100%).

Other combinations are possible, like Red/Green for instance. Results are quite the same, but obviously, you

by Francois "Coyboyt" Grassard

have to use the right model of glasses to clearly « extract » each image. Using this technique, colors of the original image can't be reproduced perfectly. Most of the time, I personally prefer to convert each image to grey scale before the tinting process.

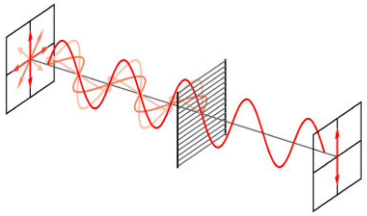
The visualisation of the « 3D effect » will be far better. Remember that today, Analglyph is not really a broadcasting choice. It's mostly used for previewing how 3D works. For instance, I don't have any 3D display at home yet. But all the studios for whom I have worked have many active or passive screens inside. When I work at home, I check my RGB color and general aspect of the render in 2D.

On the other side, I generate a grey scale anaglyph render that clearly show me the « 3D effect ». Once everything looks OK, I generate another kind of render named « Side by Side » (we will describe this later), put it on my USB key and watch the resulting sequence with the studio's screens.

So, even if Anaglyph is not a definitive solution, it can be a good introduction to the 3D world because it's really cheap ! Anaglyph glasses usually cost only from 1 Dollars/Euro to about 8 Dollars/Euro for the most « sophisticated » model (in plastic, with real glass frames). If you wanna give 3D a try inyour productions, anaglyph will probably be your best friend at start.

Polarized : You probably learned at school, that light can be explained by two kinds of phenomena. Particular, using photons concept and by spectral waves. Lightwaves are sinusoidals can exist at different frequencies. Each frequency, also know as Wavelength, represent a different color. To understand what polarization is, just take a piece of paper and draw a sinusoidal wave on it.

Now, hold the paper in your hands and turn-it in all directions. If you drop the paper on a table, this sinusoïdal wave is totally parallel with it ... and with the ground too. Orientation of this wave is now horizontal. Now, put your paper on the wall. The orientation of that wave turned by 90 degrees. Now, this orientation is vertical. When you turn on a light, billions of waves are generated in all directions and with random orientations. But you can put a filter just in front of the light to only keep waves that are horizontal or vertical. This process is called polarization.


Polarization process

By this way, you can project on the same screen and at the same time, two different images, apparently blended together, but that can be easily separated using the same kind of filter, right in front of your eyes. Filter for left eye will only let horizontal waves pass through it, and the other filter, dedicated to right eye will let only pass vertical waves. By this process, the color limitation of anaglyph images can be resolved. The other good thing about this technique is that polarized glasses (also know as passive glasses) can be produced at a really low price. But theaters who project 3D movies need two synchronized projectors (that's the case for the IMAX 3D system), each with a different kind of filter in front of them to generate the two polarized images, horizontal and vertical.

by Francois "Coyboyt" Grassard

You can see some images of IMAX 3D projectors at http://widescreenmovies.org/WSM11/3D.htm

Sequential : The third technique to be used use another kind of glasses named « active glasses ». As we describe before, the goal of the sequential files is to project the two images one after another and hide the eye that doesn't have to see the other. Using this technique, only one projector is needed, usually a digital projector like the ones from Barco or Christy, linked with a Digital Cinema Player, for instance, systems from Doremi that I recently used.

By this way, film is not needed anymore. The movie is uploaded into the player via a simple USB port and encoded using JPEG2000 for video (at a resolution of 2K or 4K) and AC3 or 6 separated waves for the audio. Both streams are packed into a single MXF file and followed by four XML files used by the player. There are five files create something called a DCP (Digital Cinema Package) and grouped into a single folder. In this MXF file, images are store sequentially. Left / Right / Left / Right / and so on.

Active LCD glasses

When I started to work with for the Doremi player, I was really surprised to read into the documentation that the embedded system was a small Linux and the player was built around FFMPEG ! Yes, when you go to a digital theaters and watch a 3D movie, FFMPEG is in the place ! Funny isn't it ? Ok, do you wanna know something even funnier? Last month, I was working for one of the biggest french TV and one TD gave me a DCP of a stereoscopic movie trailer. The first part of my job was to extract what is called essences (or streams) to make some modifications on it.

I tried to extract them using all kind of software that was installed on my computer, from Adobe, Avid, even Final Cut on a Mac next to me … no one was able to read this damned MXF ! Suddenly, I think about FFMPEG inside the Doremi player, and my poor brain made the leap to Blender. I decided to give it a try, and … YES !!! Blender can read directly an unencrypted MXF file at 4 K from a DCP right into the Sequencer. That's incredible !

Ok, I just saw two problems that, I think, can be easily corrected by the teams of FFMPEG and/or Blender (Hey devs …I love you, you know). Color space inside the DCP is not RGB but X'Y'Z'. So, color space has to be converted before displaying the movie. But I read somewhere on a roadmap schematic that color management is on the TODO list. So … I cross my fingers. OK, second problem is more touchy. In this kind of MXF, time code for Left image and Right image seems to be the same.

And when you play the video using the shortcut ALT+A into the sequencer, this one doesn't seem to base the playback on time code. For instance, when you put a 30 seconds DPC/MXF file on the timeline and you scrub along it using you mouse, you can see the end of the movie at the right time.

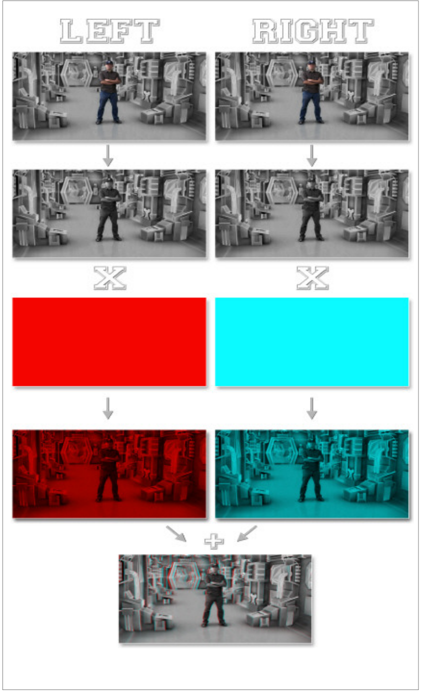
Because you don't play the movie continuously. You jump from one position to another in a random way and Blender probably looks at the time code at this time. My goal was to extract all frames of the movie in the same order they are packed into the MXF file and convert them into PNG files. I'll separate each eye later with sequencer or another editing software.

by Francois "Coyboyt" Grassard

But if you render this clip just placed on the timeline of the sequencer from image 1 to 720 (that is 30 seconds with a FPS of 24), Blender finally renders only the half of the clip while it's seems to be finish on the timeline. I guess it's because the clip was read at a frame rate of 24 FPS. And remember, when you work with sequential file, you have to double the frame rate ! When I saw the properties of the MXF clip into the sequencer,

Blender showed me that the Frame rate is set to 24 FPS. Because it simply read the meta-data store inside this MXF. But that meta-data lies to Blender ! Shame on it !!! And unfortunately, in Blender you can't change, for now I guess, the frame rate of the clip, directly in the properties. It could be really useful to avoid this kind of problem! Blender could be the first editing software that could handle DCP package!

And if one day Blender is able to produce directly a DCP package through the render panel like « OpenCinema Tools » (http://code.google.com/p/opencinematools/)... I'll buy champagne for all the devs !!! (hummm … at Blender Institute only, OK?) So, to finish on Sequential files, the worst part is that active glasses is more costly and heavier than passive ones. If you want do investigate on the DCI standard (Digitial Cinema Initiative), just go the the DCI website : www.dcimovies.com



The process of creating an anaglyph image

B.2) Difference between broadcasting technique in theaters and on 3DTV

Ok, now we know what kind of process are used to project 3D movies on the big screen, what about 3DTV ? The response is quite simple. That's exactly the same techniques … with some tiny differences. First, you have to know that most of the 3D shows are generally created at least in FullHD at a resolution of 1920x1080, square pixels.

- **Anaglyph** : Anaglyph will never be really used to broadcast 3D contents to the mass. Now that more sophisticated techniques exists, anaglyph is used for previewing only and for some « markentting experience » like Google Street view in 3D.

- **Polarized** : Same technique as the one used in theater but with a little difference. Passive screen, as every HDTV, has a resolution of 1920x1080. But in this particular case, one line on two are polarized horizontally and the others are polarized vertically. It's exactly the same as fields rendering. So, if the vendor of the screen doesn't choose to double the number of lines (to reach 2160 lines), the resolution of each frame is divided by 2 vertically. Taking this limitation in account, the resolution of an image is 1920x540. For now, I never saw a public screen with a vertical resolution of 2160 lines … but once again, I cross my finger to see it quickly.

by Francois "Coyboyt" Grassard

- **Sequential** : For now, only BluRay 3D discs can handle this kind of stream at home. The biggest advantage of BluRay is that the movie can be played at the original frame rate, generally 24 FPS, avoiding any telecine processes (even if Gamma corrections are still to be done). So, in the case of 3D movies, the frame rate will be at least 48 FPS (remember, two eyes at 24 FPS each). But you have to know that active screens have to match a minimum refresh rate to work well with 3D movies.

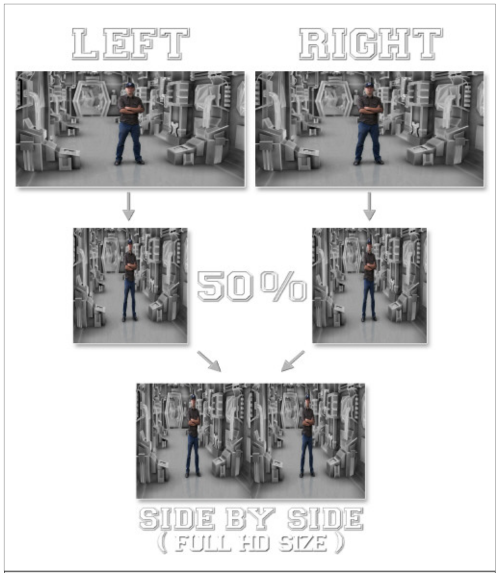
As we said before, an MXF file inside a DCP store images Left/Right/Left/Right &hellip; in this order. But if you project each image only one time, you'll probably see a flickering effect due to your retinal persistence. In theaters, even for 2D movie, the same image is shown 3 times before switching to the next one.

So, for a 3D movie, the real display sequence is Left 1 / Right 1 / Left 1 / Right 1 / Left 1 / Right 1 / Left 2 / Right 2 / and so on. So, if you quickly calculate the resulting frame rate for a BluRay 3D disc : 24 x 2 x 3 = 144 FPS/Hz. That's why your 3DTV has to have a minimal frequency of 150 Hz to comfortably display a 3D movie.

B.3) How to transport 3D streams into classical HD pipeline :

For now, there's not any 3D broadcast standard to send to the mass a FullHD 3D program at 48 frames per second through satellite, digital terrestrial service (called TNT in France) or IPTV (TV by ADSL). Until standards exist, broadcasters are constrained to use existing HD pipelines to stream 3D contents. Like they put 16/9 images into 4/3 pipeline using anamorphic images for SD, the two HD images for each eyes are squeezed to fill the space of only one HD image. Each view (left and right), are scale at 50% of there original size horizontally and place « Side-By-Side » (how became an official technical word) to create a Full HD image with a resolution of 1920x1080 containing the two views.

All programs broadcasted since 3DTV came out, including the last Soccer World Cup are done like that. Side-by-side (also know as SBS) is kind of a « first introduction » to 3D broadcasting … but horizontal resolution (and of course, details) of each image is divided by 2. Several other combinations exist (image has been darken for better visual understanding) :



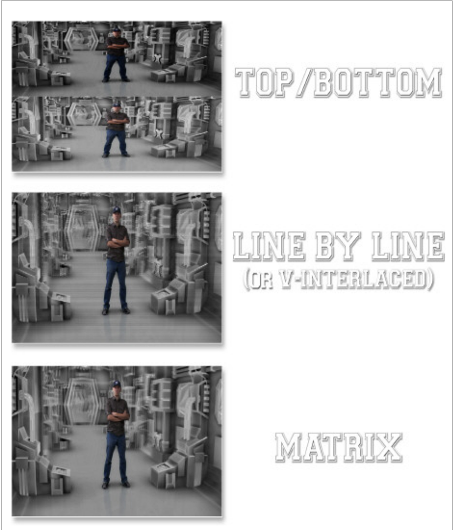
The process of creating an « Side-by-Side » image »

by Francois "Coyboyt" Grassard

- **Top/Bottom** : Same as Side-by-Side but here, the scale of 50% is done on the vertical axis.

- **Line by Line** : Nearly similar to fields, one line on two contain the left image and other lines contain the right one. More suitable to decode two images nearly at the same time and keep them synchronized if the decoder doesn't have a big buffer memory to keep one image while the second is decoded (that's the case with Side/Side or Top/Bottom techniques).

- **Matrix** : Here, pixels of each image are alternated one on two and create visually a kind of a grid. Once again, the goal is to decode the two images exactly at the same time and more precisely than the « Line by Line » technique.



Three different 3D broadcasting techniques

For now, Side-by-Side is the most used technique and all 3DTV are able to understand it and extract each images from this « composite image ».

When 3DTV receives these kind of images, there two solutions :

- **Active screen** : Each image is stretched back to their original size to produce a kind of HD image (faked by bi-linear interpolation) and are played one after another as we described previously.

- **Passive screen** : Each image is stretched back to there original size to produce a kind of HD image and played at the same time but at half resolution vertically, alternate line by line and differently polarized, as we said previously.

So, as you can see in both case, images are stretched back to their original size. In the case of active screen, we can consider that the Side-by-Side techniques reduce by 50% the horizontal resolution of original footage. But with passive screen (that doesn't have the doubled number of vertical lines yet), the vertical resolution is divide by 2 once again.

So, at the moment I wrote this (everything evolves very quickly), passive screen only shows a image that has only one quarter of the original resolution ! So for now, 3D HDTV are not always really ... HD. There will be only when broadcasters will be able to stream 2 x FullHD footage without any anamorphic tricks like Side-by-Side or somewhat.

C) Creating 3D contents using Blender 2.5 (at last):

Yes !!! Finally, here it is. After all that technical stuff, you should acquired all the knowledge to understand what we are going to create and how create it ! I know it wasn't the funniest part, but now I can directly use some terms like anaglyph, polarized or side-by-side without having to explain them, only focusing on the Blender part.

by Francois “Coyboyt” Grassard

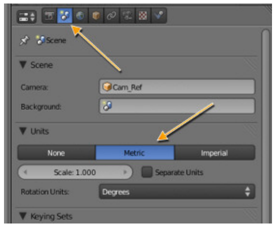
All I'm going to do will be done without the help of any script. My first goal is to describe all the process to made it easily understandable to finally, I hope, inspire more people to create scripts to automate some tasks. I already start writing scripts but at this time, I'm waiting for the 2.5 API to be fully stabilised to continue. The build of Blender 2.5 used here is r29308 take from graphicall.org.

At the end of this article I'll made some proposals to enhance Blender 3D capability and made all the work flow easier. OK ... first, let's talk about the 3D Camera rig.

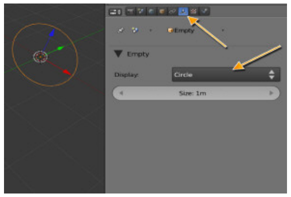
C.1) Creating the 3D Camera Rig :

As usual, if you want to work fast, you have to create some handy tool sets. As an animator has to create a rig to animate his character, we have to create our own kind of 3D camera using traditional Blender's techniques. And Blender 2.5 has some new powerful features for that. Let's see how to :

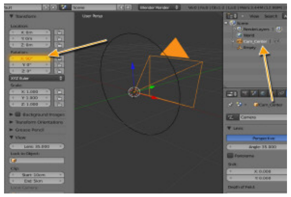
1 As we said before, taking in account the scale of your scene is really important to achieve realistic effects when you have to work with common objects used in real life. With Blender 2.5, we can now set the units system to « Metric » in the scene panel, on the right of the interface. It will be especially handy when we will set the IPD value, expressed in millimeters.



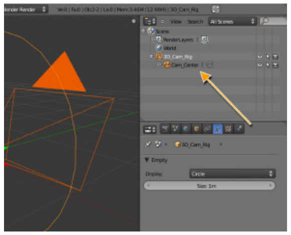
2 Now, create a simple Empty by pressing Shift+A ›› Empty and press Alt+R then Atl+G to remove any transform to replace it at the center of the world. To easily catch it in the scene, I switch to the Object Data panel and change the Display to Circle



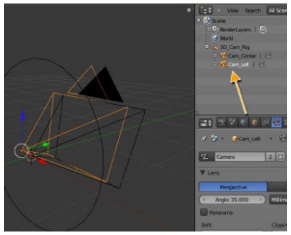
3 It's time to add a new camera to your scene by pressing Shift+A ›› Camera. Once again press Alt+R then Atl+G to replace it at the center of the world, and turn it by an angle of 90 degrees on the X axis. Via the Outliner on the top right, Ctrl+clic on Camera to rename it Cam_Center.



4 In the same way you made it for the cam, rename your Empty 3D_Cam_Rig. Select your camera, then press Shift and click the empty to add it to the selection list. With mouse cursor over the 3D View, press Ctrl+P to Set parent to object.

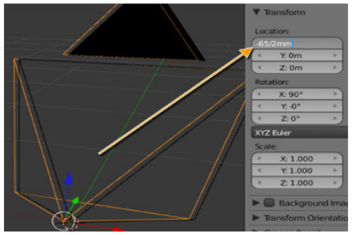


5 Select Cam_Center and press Alt+D to create a linked copy of that camera. Rename the duplicated one Cam_Left. As you can see, if you change the Angle value of Cam_Center , controlling the Field of View, the FOV of Cam_Left changes in the same way and exactly at the same time. All parameters are fully linked.

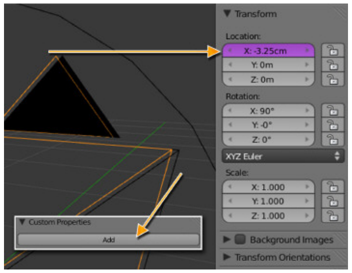


by Francois "Coyboyt" Grassard

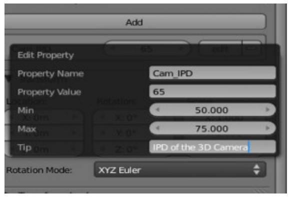
6 Select Cam_Left then press Nkey the show the Transform panel, on the right of 3D View. Look at the first Location parameter shown, named X. You can type any kind of value in this field, and because you previously switched your scene unit to Metric, you can enter a value followed by mm, as Millimeters. If your value is positive, the camera moves to the right of the Cam_center. If it's negative, it moves to the left. So, because your duplicated cam is named Cam_Left, the value for X will probably be negative in the Local Space of it's parent, the Empty. As we previously said, the most used IPD is around 65mm. But you have to divide this value by two because the left cam will moves by 65/2 on the left, and the right cam will moves by 65/2 on the right. So, you can directly type in the field -65/2mm. Magic, isn't it ?


Transform
Location: -65/2mm, Y: 0m, Z: 0m
Rotation: X: 90°, Y: -0°, Z: 0°
XYZ Euler
Scale: X: 1.000, Y: 1.000, Z: 1.000

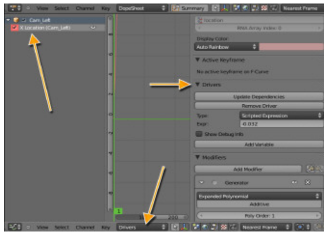
7 Ok, now we have understood how this property works, right-click on the Location X value and choose Add single Driver. This field is now purple colored, meaning that it's controlled by a Driver. Now, select the Empty named 3D_Cam_Rig and switch to the Object panel. Scroll down the panel to reach Custom Properties. For me, that's one of the most exciting features of Blender 2.5. The ability to add an unlimited number of custom values to control other parameters. All kinds of parameters on every object! Expand this panel and click the Add button.


Transform
Location: X: -3.25cm, Y: 0m, Z: 0m
Rotation: X: 90°, Y: -0°, Z: 0°
XYZ Euler
Scale: X: 1.000, Y: 1.000, Z: 1.000

8 A new property is now created. All sub-panels can be easily moved across the Properties window by simply clicking and dragging it's name. I suggest to drag the Custom Properties to the top. By this way, you can see all controllers of your rig when you select it. For now, the new property is named prop. Click the edit button to change this name to Cam_IPD. Because the IPD of human is considered to be in a range of 50-75mm (remember, this article is not for aliens audience), set min to 50, max to 75 and Property value to 65 who is a medium value. If you want, you can fill the Tip field with IPD of the 3D Camera.

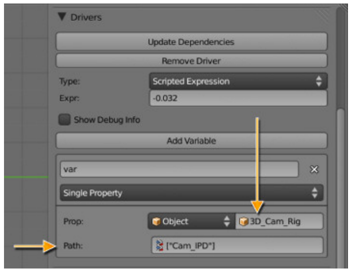

Edit Property
Property Name: Cam_IPD
Property Value: 65
Min: 50.000
Max: 75.000
Tip: IPD of the 3D Camera
Rotation Mode: XYZ Euler

9 Now, right click on the 65 value and choose Copy Data Path. In Blender 2.5, each Datablock, as custom property is, can be identified by a unique ID named Data Path. Switch to workspace named Animation then select Cam_Left. The lower Graph Editor is set to F-Curves Editor. Click on that button to switch to Drivers. The X_Location(Cam_Left) property appear on the left. Let your mouse cursor over the Graph Editor and press Nkey to display the properties panel on the right of the editor.



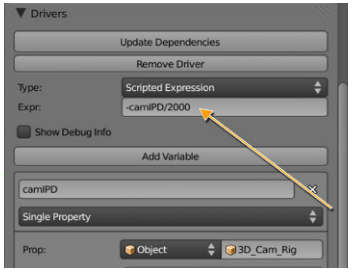
by Francois "Coyboyt" Grassard

10 In the Drivers panel, click the Add Variable button then click the empty field, just next to the Object drop-down and choose 3D_Cam_Rig. A new field named Path is now shown. Click on the empty field and press Ctrl+Vkey to paste the Data Path you previously copied from the Cam_IPD parameter. You should have something like ["Cam_IPD"].



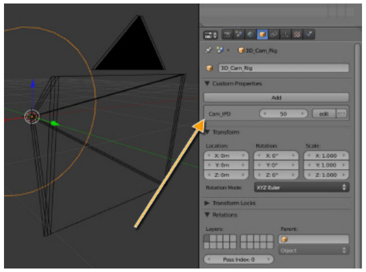
Type: Scripted Expression
Expr: -0.032
var
Single Property
Prop: Object 3D_Cam_Rig
Path: ["Cam_IPD"]

11 ["Cam_IPD"] is now connected to this new variable named var. Change it's named to camIPD. Just over the Add Variable button, you can see the field named Expr. This one is the final output of the driver, directly plugged into the X_Location of Cam_Left. So, if you simply type camIPD in this field, X_Location will have exactly the same value as the Custom Property.In the over case, you want to create 3D elements that only live in the wonderful Blender CG world ! In this case, knowing where the focal plan is (where each direction of cameras intersect) is really difficult and it could be useful to control the position of this focal plane only using a Empty. So, we have to create a kind of mixed setup, suitable for each case. To do that, we have to add new Custom Properties to 3D_Cam_Rig named FP_Influence with a min/max range of 0-1 and another one named Vergence with a range of 0-5, even if 5 is probably to high. Remember, vergence value are usually between 0-2 degrees to avoid incredible headache.

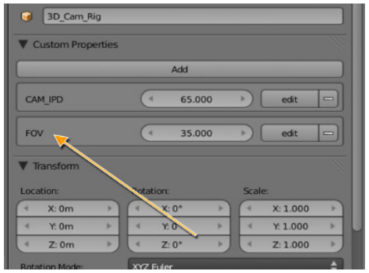


Type: Scripted Expression
Expr: -camIPD/2000
camIPD
Single Property
Prop: Object 3D_Cam_Rig

12 Once the Cam_Left is set, just select it and press Alt+D to create a linked copy. Rename it Cam_Right. Even if all parameters of the camera are fully linked to the original on,e expression typed in the driver settings seems to work like a non linked copy, that is a really good think for us in this case. You just have do delete the minus sign in front of the expression : camIPD/2000. And that's it ! Your IPD constraint is set.



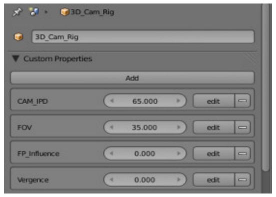
13 Using exactly the same technique, you can add a new Custom Property to your 3D_Cam_Rig, controlling the FOV of Cam_Ref. Because Cam_Left and Cam_Right are linked copies of this original object, their respective FOV will change too.



3D_Cam_Rig
CAM_IPD 65.000
FOV 35.000
Location: X: 0m Y: 0m Z: 0m
Rotation: X: 0° Y: 0° Z: 0°
Scale: X: 1.000 Y: 1.000 Z: 1.000
Rotation Mode: XYZ Euler

by Francois "Coyboyt" Grassard

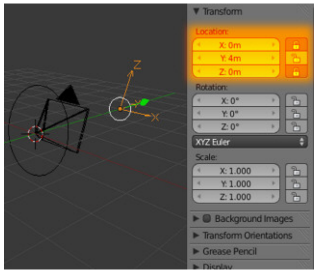
14 OK, now let's talk about vergence. As we said before, you can shoot your scene using a fully parallel setup or a convergent one. Sometimes, you have to set vergence using an angle value, for instance when you have to put a CG element into a live action footage, shot in 3D with a vergence of 1 degree.


3D_Cam_Rig

Custom Properties

CAM_IPD	65.000
FOV	35.000
FP_Influence	0.000
Vergence	0.000

15 Create another Shift+A ›› Empty, then press Nkey to display the « Transform » panel on the right of the 3D Viewport. Set the XYZ Location values to 0/4/0 the lock X and Z parameters. You can now simply move that Empty using Gkey to move it away from the camera. Rename that Empty FocalPlane and parent it to 3D_Cam_Rig.


Transform

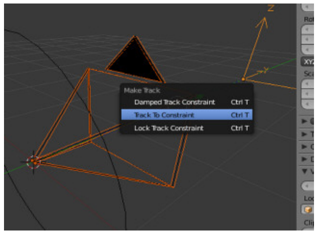
Location: X: 0m, Y: 4m, Z: 0m

Rotation: X: 0°, Y: 0°, Z: 0°

XYZ Euler

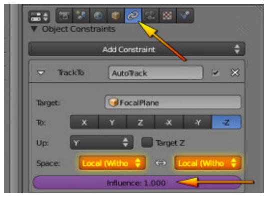
Scale: X: 1.000, Y: 1.000, Z: 1.000

16 Select Cam_Left, Cam_Right and finally FocalPlane. Press Ctrl+T and select Track to Constrain. Now, if you move FocalPlane, you can see the vergence of cameras change. By moving this FocalPlane, you can easily choose which element of your 3D world is located at the distance of the screen, what is behind it and what's popping out. of the Empty.


Make Track
Damped Track Constraint Ctrl T
Track To Constraint Ctrl T
Lock Track Constraint Ctrl T

But … remember. IPD value have to be divided by two, and one unit in Blender world is equal to 1 meter, because you previously set unit to Metric. The Custom Property added to the Empty is designed to work in millimeters. So, you also have to divide the camIPD by 1000. Finaly, the result is (camIPD/1000)/2 = camIPD/2000. But don't forget to inverse the result because the left cam has to move … on the left : The expression to enter in the field is : -camIPD/2000

17 If you select 3D_Cam_Rig and try to turn it by pressing Rkey twice, you can see that Cam_Left and Cam_Right don't rotate in the same way as Cam_Center. To fix this bad behavior, you have to switch to Local Space from the two drop down menus. Right click on FP_Influence and choose Add Driver.


Object Constraints

TrackTo: AutoTrack

Target: FocalPlane

To: -Z

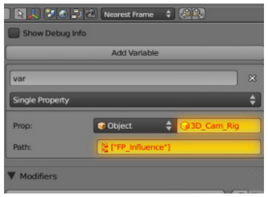
Up: Y

Target Z

Space: Local (Witho ⇔ Local (Witho

Influence: 1.000

18 Switch back to the Graph Editor displaying Drivers. Select Influence on the left of the window and switch type from Scripted Expression to Averaged Value. Click Add variable button, choose 3D_Cam_Rig just next to the Object drop down. Right click the FP_Influence parameter of 3D_Cam_Rig to Copy Data Path and paste it to Path. Now, you can control the influence of the Track to constrain using your Custom Property. By setting FP_Influence to 0, your 3DCam rig will be parallel. If FP_Influence is set to 1, the rig will be convergent. Just do the same for Cam_Right.


Nearest Frame

Show Debug Info

Add Variable

var

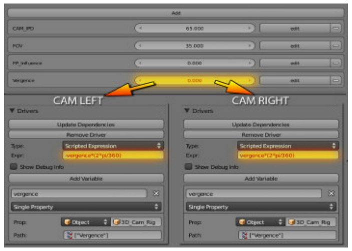
Single Property

Prop: Object 3D_Cam_Rig

Path: ["FP_Influence"]

Modifiers

by Francois "Coyboyt" Grassard



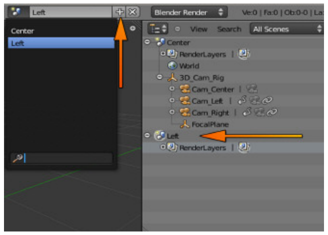
19 Finally, as you have previously done for other parameters, create a Driver for Rot Z of each Camera and connect them to Vergence parameter of 3D_Cam_Rig. But this time, you have to convert the rotation values from Rot Z, which is expressed in Radians, to values in degrees. For Cam_Right, this value has to be positive, for Cam_Left it has to be negative.

Your 3D camera is now completely ready to use. Don't forget, if you want to control vergence using a angular value, to set the FP_Influence to 0. You can even have a mixed setup using a value between 0 and 1. Of course, this rig is only a base. For instance, to create a two points camera, using target, you just have to add a new empty and link it to 3D_Cam_Rig using a Track to constrain. Keep in mind that 3D_Cam_Rig can be consider as a single camera.

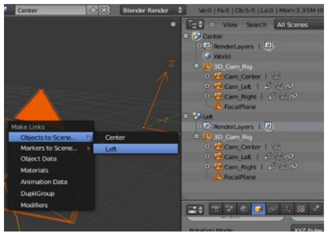
To layout your shot, simply use Cam_Center and check from time to time what append to Cam_Left and Cam_Right.

C.2) Set the Left and Right scenes and compositing nodes :

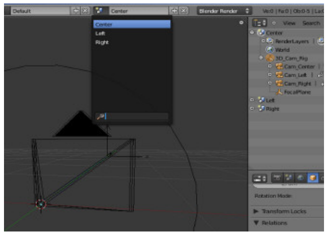
To separately render each camera, you have to create more than one scene. For now, each scene in Blender uses the camera tagged as Active to render. You can't have two active cameras at the same time. Otherwise, Blender wouldn't know what camera to use, that's logical. So, to render multiple camera views in a single render job, you have to add two more scenes. Here, we gonna explain how to make an anaglyph render using there two views.



1 At the top of the interface, rename the current scene to Center, then click the « + » button to add a new one, named Left. As you can see in the Outliner, Left scene is totally empty when it's created.



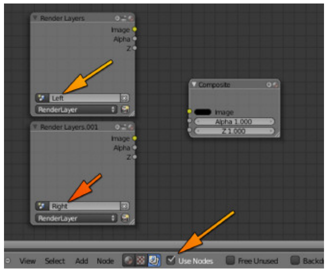
2 Switch back to Center scene then press Akey to select all objects. Press Ctrl+Lkey (L as Link) and choose Scene ›› Left. Look at the Outliner, Left scene is filled with the same objects as the Center Scene. It's important to notice that all objects are linked together and not copying. Any modification done in Center scene will be done in any other scene.



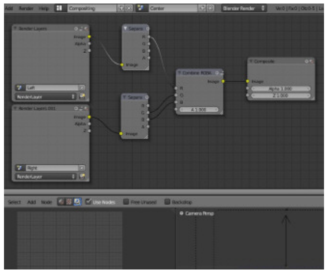
3 Repeat the two last steps to create another scene names Right and link all objects to it. Now, jump into the Left scene, select Cam_Left and press Ctrl +Numpad 0 to set this cam as the active cam. Do the same for the Right scene to set Cam_Right as the active cam, and finally Cam_Center for Center scene.

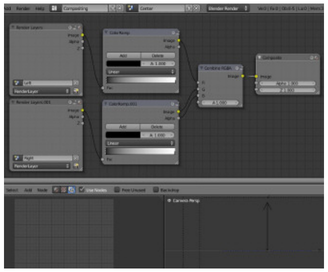
by Francois "Coyboyt" Grassard

4 Switch back to Center scene then jump into the compositing workspace. Click Use nodes on the bottom of the node editor. Render Layers and Composite nodes. In the first one, you can choose which scene Blender will render. Choose Left, then select that node and press Shift+Dkey to duplicate it and set the duplicated one to Right.



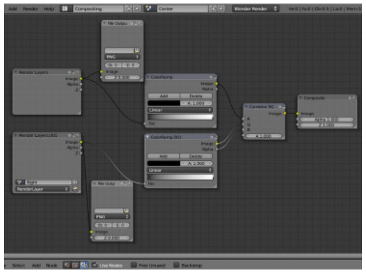
5 As we previously said, many solutions exist to broadcast 3D images. We're going to describe here the most simple and suitable for everyone that doesn't have a 3D screen : Analgyph. Add to your compositing graph two Separate RGBA nodes, one for each Render Layers node. Add a Combine RGBA node and plug the R output to the Separate RGBA R input node connected to Left render. Plug the B and G output to the Separate RGBA B and G input.





6 As we previously said, this kind of Analyph combination tries to keep some information about color, but it's never really works. To achieve a good representation of the « 3D effect », you have to turn each render to grey scale with a default Coloramp before combining them. So, the two Separate RGBA nodes can now be deleted.

7 It's always good to keep the original renders on the disc before combining them. To do that, you can add two File Output nodes for each render. One thing you have to know : even if you want to only output each render to work later with them, you have to combine them into one Composite node. Even if it's a simple Color Mix you don't care about. Otherwise, only one render will be launched.



We can't describe here all the techniques to generate other combinations like Side by Side, or V-Interlace or whatever. But you'll find in the Blend files provided with you rfavorite magazine a .blend that combine Anaglyph / Raw output / Side by Side in the same render job. It will be certainly useful for some people.

More info about render settings. Remember that each scene share the sames objects because they're linked together. But each scene has their own render settings. Anti-aliasing, ambient occlusion, and so many parameters can be different in each scene to optimize render time. But most of the time, you will have to set all parameters 3 times.

A good idea for python fanatic could be a function to duplicate scene parameters for one scene to another. It could be really useful.

by Francois “Coyboyt” Grassard

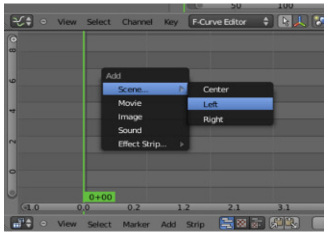
The Add-on system is so good with Blender 2.5, all is now possible.

And don't forget, if you add a new object in the Center scene, you have to link it with the two other scenes with Ctrl+Lkey. Once again, a magic button « Link them all » could be useful. ;o)

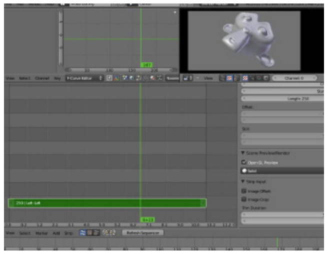
C.3) Previewing 3D images in real time using sequencer :

Now you know how to render an anaglyph image with a maximum control over your 3D settings, I'm going to explain a really good trick to preview the « 3D effect » in real time, while you are working on your scene. Finding the right distance of a object in 3D if often a really time consuming task. With analgyph glasses on your nose and real time anaglyph preview, checking that distance is incredibly easy. Let's see how to :

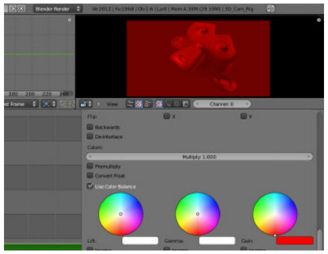
1 Jump into the Video editing workspace, and press Shift+A ›› Scene. Here you can choose between Left and Right. Start to choose Left and add that clip on the timeline.



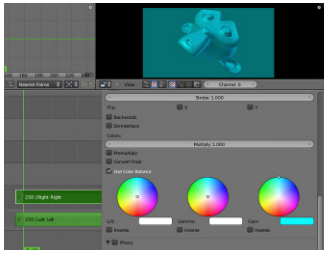
2 Select the Left clip and look at there properties on the right of the screen. Scroll down to Scene Preview/Render, check Open GL Preview and choose Solid in the Drop-Down menu, just below. You can now, by moving the timeline, see your animation in real time ! That's one of the Blender 2.5 benefits ... one of the best for me !



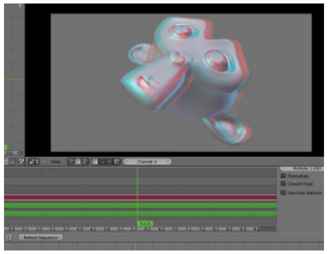
3 Scroll down once again in parameters to check Use Color Balance. Once checked, three wheels appears with colored squares bellow. Click on the right one and set RGB color to 255/0/0.



4 Using the same method, add on a second track a new clip for the Right scene. Once again, check the Open GL Preview and use Color Balance to change his color to Cyan (RGB=0/255/255).



5 Select the two clips then press Shift+A ›› Effect Strip ›› Add. And here is your analgyph render. Just wear your Red/Cyan glasses and you'll see the 3D Effect ... in real time ! Since Blender 2.5, it's possible to open a window with only the sequencer monitor. So, you can work on a classic 3D view and see the result in a Sequencer monitor, just next to it !



by Francois "Coyboyt" Grassard

During a production I worked on last month, I used the same technique to generate a Side-by-Side render using Transform effect strip and some Metastripes. I plugged the really expensive 3D passive screen of the company as a second screen (as extended desktop) on HDMI port and 1920x1080 resolution. On that second screen I placed a Blender window with a Sequencer screen and maximized it (Alt+F11).

I removed any header to obtain a 3D HD preview in real time of my scene using professional 3D screen ! Just two bad things remain. The « + » sign can't be removed, like the 3 stripes to divide the screen. It can be a little disturbing. If devs hear me, could it be possible to have a totally empty full screen mode ? ;o) Thank you in advance, guys !!!

D) How can we improve Blender 3D workflow :

All we have done during this article is done with built in Blender features. As a conclusion, I'd like to make some humble proposals to every devs that would like to improve Blender stereoscopic capabilities. Of course, if any developers from the Blender Institute read this article, these proposals are primary for you, but not only. With the wonderful « Add-On » support, anybody can work an play around to improve there functionalities. So that's a non exhaustive wish list … but I think one or two of there proposals could be interesting for the community. At least, I hope. ;o) Some of them have been already discussed in this article.

Possibility to directly create a 3D build in camera, with same controls created in our 3D_Cam_Rig. For instance Shift+AKey >> 3D Camera. Nearly similar as the camera that can be found in Eyeon Fusion, for instance.

As a consequence, RenderLayer in compositing could have two outputs, Left Output and Right Output

These two outputs could be plugged into a new compositing node specially created to directly generate Side-By-Side, Anaglyph, Line-by-Line, Matrix or Top/Bottom.

Render layer node could output a quick render directly took from the OpenGL View (antialiased if possible, and not only forced by FSAA the graphic cards), as the Scene Clip in the Sequencer. Using « Baked textured » ;, we could render very quickly a sterescopic view of a virtual set and composite over them a chromakeyed human (we will discuss about this probably in BAM 29 ;o)

It could be really useful to link two clips in Sequencer. Each modification on one clip (left view) could be reported on the other clip (right view). I know it can be already done using metastrip, but in some cases, using separated clips is better.

I don't know if it's possible, but Raytree could be only computed once for two views, because they are nearly the same.

The best feature that can be added to Blender regarding 3D Workflow : Anaglyph preview directly into the 3D View to avoid the trick using the sequencer. We could directly see in real time the « 3D Effect » during the layout. BGE already provides this feature.

- A real full screen window, without any « + » or « separation strips » to send to a second screen a side-by-side image, displayed by a 3DTV, plugged as a second screen via a HDMI port.
- Color management and colorspace conversion : RGB >> YUV, RGB, X'Y'Z', and so many more … ;o)
- Fix issues about frame rate in DCP / 3D MXF reading as described previously.
- Directly render as side-by-side, using for instance a new sequencer Effect Stripe.

by Francois "Coyboyt" Grassard

And so many more … ;o)

So many things could be explored, like support of disparity maps to help rotoscoping tasks. For instance, right eye's render could be encoded only by difference regarding left eye's render. With this totally loss-less process, file's weight could be reduced by around 40% !

I hope this article was for you a good introduction to the 3D world and gave you inspiration to do more 3D with Blender 2.5. You'll find in a zip file a lot of scenes to help you understand how to use 3D_Cam_Rig and how to create good and spectacular 3D contents. See ya … in 3D ■

I Love My Bubble

By Max Kielland

Introduction

I usually browse the net and admire other artist's works and to get inspiration. Not only inspiration for subjects or scenes, but also the enthusiasm you will need to break down a scene and actually pull it off in Blender.

This time I wanted to create a sugar sweet fluffy effect with bubbles floating in a cloudy atmosphere, so be prepared for some pink!

We will use the particle system, compositor, UV wrapping, some textures and animation. There will also be some useful tips on workflow and tools. Blender 2.52 is still somewhat buggy so you might run into some strange behaviour. Remember to save [CTRL+S] often!

Due to a bug (?!?) the particle system will not animate right sometimes. When this happens it can usually be fixed by going to frame 1 and then re-enter the particle system's start frame.

The setup

When I started to use Blender, I tried to use the strict 4 view ISO layout but quickly found it to take up to much valuable space for the tools. However I usually use a 2 split view, one for modelling and one for the camera. In this way I can immediately see if my objects are the right size and in the right place while I model them.

You can have multiple cameras for different test shots and one for the main render shot. You change the default camera by first selecting the desired camera in the outliner, position the mouse over the 3D view you want to change and then press [CTRL + Num0]. Another advantage is that you can adjust the camera in your main 3D view and at the same time see through the camera in the other view while you position it.

First delete everything in your scene by hitting [A] to select all, then press [X] and confirm to delete everything. Add a light with [SHIFT+A] and select Lamp>>Sun, name it to Main Light. I prefer to have my main light a little bit stronger so go into the Light window and in the Lamp panel change Energy to 1.16. Now add a camera with [SHIFT+A] and select Camera, name it to Main Camera.

In all my default projects I have one sun light and one camera already rigged. I also have both the camera and the light to track a target. In this way I can move around both the camera and the light and always be sure to have my target well lit and in camera view.

To do this we first create a target object with [SHIFT+A] and select Empty. Everything you add to the scene will be located at your 3D cursor. So if your 3D cursor isn't at position 0,0,0 you can easily change that from the Transform panel properties window. Toggle the properties window with [N], pointing the mouse over your 3D view. Under the View panel's 3D Cursor setting, you can set the position to whatever you like (0,0,0) in this case.

If your empty ended up in the wrong place, don't panic! In the same window (or in the object window) you can enter the exact coordinates in the Transform panel for any selected object in your scene. Now make sure your empty is located at 0,0,0 and name it Camera Target under the in the Object window .

By Max Kielland

Since everything now has ended up in the same spot it is an excellent opportunity to exercise the outline window. Use the outline window to easily select objects by their names.

In my scene I placed the Main Camera at X 0, Y -19, Z 0 and the Main Light at X -6, Y -16 Z 15. You can enter these coordinates directly in the Transform panel under Location. There is no point in changing the Rotation because the Track To modifier we will apply next will override the rotation.

Since we removed the camera before, our little view to the right needs to be set back to Camera view. Hold the mouse over the window and press [Num0] to change to the active camera view.

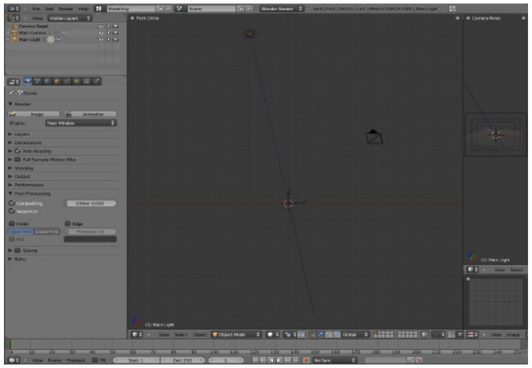
Select the Camera and go to the Constraints window . Open up the Add Constraint and select Track To. As Target select our Camera Target and watch the camera view. Oops that looks a bit awkward. You need to tell the Camera what axis should point to the target and what axis is the up-axis.

Set To to –Z and Up to Y. Now your camera should point at the Camera Target empty.

Do the same for the Sun Light. Now you should see a blue dotted constraint line from the light and the Camera to the Camera Target.

My work layout looks like this:

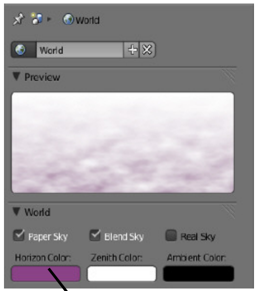
I find the Outline window very useful to quickly select an object. I have it filtered on "Visible Layers" so I only see the relevant objects. To the right you see my camera window and at the bottom the timeline window so I quickly can move between the frames. The small UV window is good for quick access to reference images, UV maps and rendered layers..



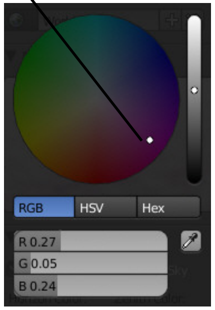
My default setup is included for download. Let's get on with this tutorial and create the fluffy clouds...

Up in the clouds

You could create the clouds with Blender's new smoke system with a great load on your CPU, or you can fake it!



Go to the World window and tick the Paper Sky and Blender Sky boxes. Then head immediately over to the Texture window and select the first texture slot. Press the new button and leave the Type as Clouds (I guess you can see where we are heading here). Leave all the other parameters as they are.



R 0.27
G 0.05
B 0.24

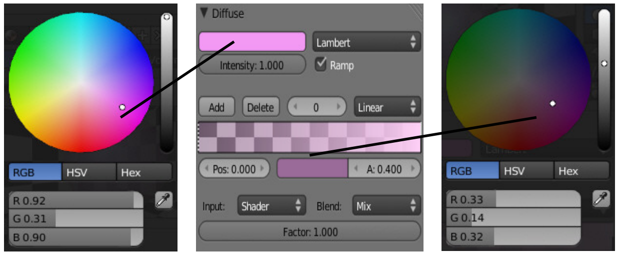
Head back to the World window and set the Horizon Color to pink, Zenith Color to white and Ambient color to black. Now you can see a sort of cloudy image appear in the Preview panel.

By Max Kielland

Since we have now created our clouds as an environment map, it takes virtually no time at all to process when we render.

Bubble trouble in paradise

We need to create a template to represent the bubbles. Press [SHIFT+A] and select Mesh>>UV Sphere, set it to smooth in the Tool shelf window (toggle with [T] in 3D view) and name it Bubble. Move it out of the way where you can easily select it. I put mine at location -10, -15, 5, scaled it to 0.113 on all axes and changed dimension to 0.425,0.25,0.425. You may need to zoom out to see it. Use the mouse wheel to zoom in and out.


RGB HSV Hex
R 0.92
G 0.31
B 0.90

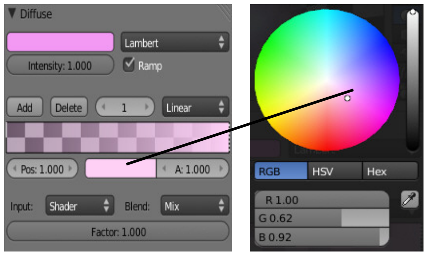
Diffuse
Lambert
Intensity: 1.000
✓ Ramp
Add Delete 0 Linear
Pos: 0.000 A: 0.400
Input: Shader Blend: Mix
Factor: 1.000

RGB HSV Hex
R 0.33
G 0.14
B 0.32

I want it to be a pink shiny bubble so we need to work on a new material. Head over to the Material window and press the new button to add a new material and slot for this object, name it Bubble Mat.

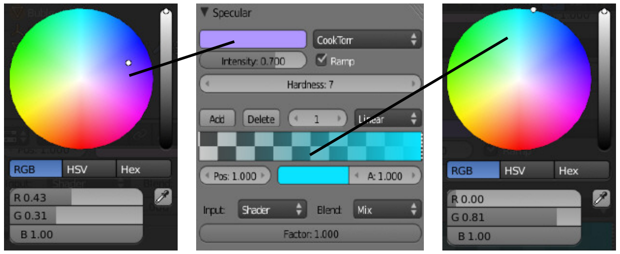
In the Diffuse panel set colour to a nice pink one, Intensity to 1.000 and tick the Ramp box.

The ramp allows us to blend in another colour depending on the amount of light hitting the surface. We can give the bubble more depth by using a ramp going from dark pink to bright pink with different alpha values. The ramp already has 2 positions by default, one on each end. At the left position (0) set Colour to a little darker pink and Alpha to 0.400. At the second one (1) set the colour to almost white and Alpha to 1.000.


Diffuse
Lambert
Intensity: 1.000
✓ Ramp
Add Delete 1 Linear
Pos: 1.000 A: 1.000
Input: Shader Blend: Mix
Factor: 1.000

RGB HSV Hex
R 1.00
G 0.62
B 0.92

Now we want the highlighting, Specular, to be a bit more blue, also with a more bluish ramp. Go down to the Specular panel and set the colour to a more pink-blue. Tick the Ramp box to bring up the colour ramp.


RGB HSV Hex
R 0.43
G 0.31
B 1.00

Specular
CookTorr
Intensity: 0.700
✓ Ramp
Hardness: 7
Add Delete 1 Linear
Pos: 1.000 A: 1.000
Input Shader Blend Mix
Factor: 1.000

RGB HSV Hex
R 0.00
G 0.81
B 1.00

We do the same here but go from a black to a turquoise colour. Leave the alpha values but change the second colour to turquoise.

A bubble is not a real bubble unless it has some transparency, so just tick the box in the Transparency panel and set Alpha to 0.400. This gives us enough transparency to still be able to see the bubble.

The last thing we will do to add more depth is to have the bubbles receive transparency shadows/light. This will shine up the opposite sides inside the bubble as well. Go down to the Shadow panel and tick Receive Transparent.

By Max Kielland

Now we should have a nice pink bubble.

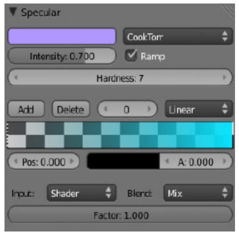
Bubbles, bubbles, bubbles...

We will create the bubbles with a particle system and for that we first need an emitter. The emitter object is sending out the particles from its vertices, faces or volume. Create a Cube with [SHIFT+A] and select Mesh>>Cube, name it Bubbles.

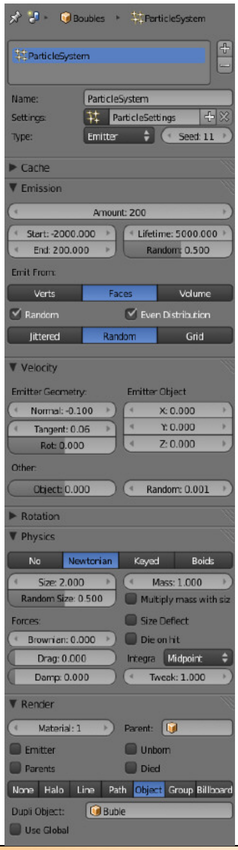
In my scene I placed the Bubbles at X 0, Y 0, Z 0 and scale X 9, Y 9, Z 9. You can enter the Location and Scale directly in the Object window but we also need to set the Dimension to 18, 16 and 10. For some reason the Dimension can only be accessed from the Properties window. Bring up the properties window with [N] and do the changes in the Transform panel.

If your box is solid you can toggle between wireframe and solid with [Z].

Since I, in the end, want to animate the bubbles gracefully floating in the clouds I need to plan how the particles are entering the scene. I only want the particles to float in from the sides and bottom-up. I also don't want them to suddenly just appear and disappear in the camera view.



Specular
CookTorr
Intensity: 0.700
Ramp ✓
Hardness: 7
Add | Delete | 0 | Linear
Pos: 0.000 | A: 0.000
Input: Shader | Blend: Mix
Factor: 1.000



Boubles ▸ ParticleSystem
ParticleSystem
Name: ParticleSystem
Settings: ParticleSettings
Type: Emitter | Seed: 11
Cache
Emission
Amount: 200
Start: -2000.000 | Lifetime: 5000.000
End: 200.000 | Random: 0.500
Emit From: Verts | [✓ Faces] | Volume
Random ✓ | Even Distribution ✓
Jittered | [✓ Random] | Grid
Velocity
Emitter Geometry: | Emitter Object
Normal: -0.100 | X: 0.000
Tangent: 0.06 | Y: 0.000
Rot: 0.000 | Z: 0.000
Other:
Object: 0.000 | Random: 0.001
Rotation
Physics
No | [✓ Newtonian] | Keyed | Boids
Size: 2.000 | Mass: 1.000
Random Size: 0.500 | Multiply mass with siz
Forces: | Size Deflect
Brownian: 0.000 | Die on hit
Drag: 0.000 | Integra Midpoint
Damp: 0.000 | Tweak: 1.000
Render
Material: 1 | Parent:
Emitter | Unborn
Parents | Died
None | Halo | Line | Path | [✓ Object] | Group | Billboard
Dupli Object: Buble
Use Global

To better understand this let us take a look at how particles are emitted (generated).

A particle system is a flow of particles over time. This means that they will begin to emit at the start frame and stop at the end frame. Between the start and the end frame every particle generated will just appear on the emitter and disappear when it dies. Did I say die? Yes, each particle also have a life time starting at the frame it is generated counting forward and when it has been in the scene for its Life Time number of frames it will just disappear.

So we will avoid having the particles being generated within our camera view and they must live long enough to not disappear in the camera view during the animation.

Let us first attach a particle system to our newly created emitter Bubbles.

Go to the Particle window and press the + button to add a new system.

Leave the Type as Emitter but change Seed to 11. The seed only tells the randomizer how to initialize the random number generation. I found 11 generates a nice looking particle flow.

Let us have a look at the Emission panel. We don't want a forest of bubbles so change the Amount to 200.

As I mentioned before we want the bubbles to float into the camera view so the idea is to let the camera view fit inside the emitter. If we then let the faces of the emitter generate the particles, they will be outside the camera view! To do this set Emit From to Faces and Random.

But we are still not seeing any particles! This is because we are still on frame 1 in our animation.

By Max Kielland

If you start/stop the animation with [ALT+A] you will see how the particles start to emit. But they look a bit small, not like bubbles.

For this we will actually not render the particles themselves, instead the particle will become an “empty” to guide another object.

Under the Render tab change from Halo to Object and select our Bubble as the Dupli Object. A referenced copy of the dupli object will be placed at each particle instead of the particle itself. Any change to our Bubble will now affect all the bubbles in our particle system. We also don’t want to render the emitter itself so untick the Emitter box as well.

As you can see they are still too small to be taken for bubbles. To get more variations in the bubble size go to the Physics tab and change Size to 2 and Random Size to 0.5.

But wait a minute; they all move in the wrong direction, we wanted them inside the camera view! Let us take a look at the Velocity panel. Here we can control the initial speed and direction of our particles.

Set Emitter Geometry Normal to -0.100 to have them float slowly.

A positive value will send the particles in the face’s normal direction so a negative value will send them in the opposite direction of the face’s normal. I guess flipping the emitter’s normals would have done the same trick, but let us keep things simple and don’t mess with Blender’s way of defining the normal directions.

Now the particles are moving inside the emitter but they aren’t moving slow, they are falling down with increasing speed... Time to get physical.

This has to do with gravity (hence the Newtonian system). We need to change the gravity to have them float.

Go to the Scene window and in the Gravity tab untick the Gravity box. This gives us zero gravity, just like in outer space. Now the small initial force from the emitter’s normals will not be changed and the bubbles will float forever with the same speed in the normal’s direction.

Now they aren’t falling down but instead they move so slow they will actually not reach the camera before the animation is done. Is there not a way to force the bubbles to flow before the animation starts? Yes there is!

Go back to the Particle window and the Emission tab again; look at the Start, End and Lifetime. Regardless of what frame our animation starts and stops, our particle system can start and stop at other frames.

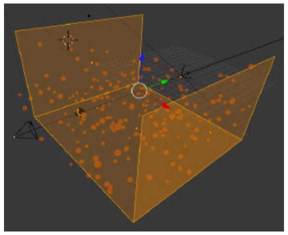
We need the particle system to start before our actual animation. In this way it will already have generated enough particles to fill the camera view when the actual animation starts. Set Start to -2000 to have the particle system started 2000 frames before the actual animation will render.

This created another unwanted side effect because the particles will only live for 50 frames and then die, they will still not reach the camera view. Change Lifetime to 5000 to ensure that they will live through the whole animation.

Still the bubbles are appearing and disappearing in the camera view and we have bubbles coming from the above, floating down. This is because the emitters back, front and top faces are emitting particles straight into the camera view. Select the emitter box and go into edit mode with [TAB]. Select the top, front and back face and delete them with [X], remove faces.

By Max Kielland

Now the camera should be looking into a corridor without a roof.



Stay focused or not...

If we render the scene now we have a lot of bubbles but we are still missing the soft cute feeling. In real world photography we can create an effect of a sharp motive with a blurry background and foreground. This will "soften" the image quite a lot and make it fluffier. To do this we need the compositor, so head over to the Compositor (aka Node editor)

Without creating a new tutorial on compositing we can briefly say that we can stream information from the rendering process through a number of black boxes (hereafter called nodes) to add or subtract data/effects from the rendered scene.

Start by ticking the Use Nodes box and Blender will create two basic nodes for you. The left node Render Layers is taking data from your scene and streams the information into the compositor through different channels. As you can see the Image channel is already connected to the Composite node's Image input. The composite node is your end station and it is at this point the final render is produced (your render window). All channels on the left side of a node are inputs and the right side are outputs.

With this setup nothing extraordinary will happen so we will add a node called Defocus. Hit [SHIFT+A] and chose Filter>>Defocus. Connect the Render Layer's Image stream to the Defocus' Image input and the Defocus Image stream (right side) to the Composite node's Image.

In the Defocus node set Bokeh Type to Circular, fStops to 13.000 and Treshold to 0.500.

Well, we still don't have that blurry effect and that is because the Defocus node has no information about where the objects are located in space and where the focus point is.

Head back to your 3D views and select the Main Camera.

In the Object data window's Display tab, tick the Limits box to see the cameras various limits in the editor.

I also prefer to tick the **Title Safe** and **Passepartout** boxes as well.

In the Lens tab, by default the Angle is set to 35.000 and it represent a 35mm lens. This gives some distortion to the perspective, just as a real camera does. Doctors and scientists have calculated the eye to be approximately a 48mm lens. So to get a little bit closer to reality, set the Angle to 48.000 millimetres.

To better visualize the next step, switch over the 3D view to a top View [Num7] where you can see the whole Emitter box and the camera.

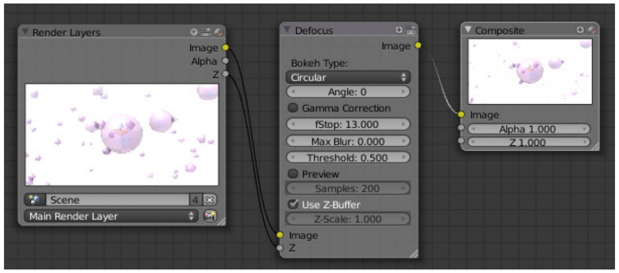
Back in the Lens tab go down to Depth of Field Distance. If you Left-click, hold and drag the mouse you will see how a little line will move along your camera track. This is your focal point! Everything at this line will be crisp and clear, in focus. Set this to 5.

But still, we need to transfer this information over to the compositor. Go to the Render window 📷 and open up the Layers tab. Under passes you will find a list of information that can be passed along to the compositor. Tick the Z box to pass the depth of all objects.

If we now switch over to the Compositor again you will notice that the Render Layers node now has a new stream: Z. Connect this one to the Defocus node's Z input. Make sure Use Z-Buffer is ticked.

By Max Kielland

If you render now you will have that blurry soft effect.



Render Layers: Image, Alpha, Z; Scene; Main Render Layer

Defocus: Bokeh Type: Circular; Angle: 0; Gamma Correction; fStop: 13.000; Max Blur: 0.000; Threshold: 0.500; Preview; Samples: 200; Use Z-Buffer; Z-Scale: 1.000; Image; Z

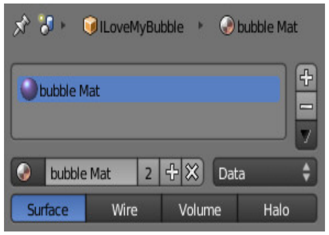
Composite: Image; Alpha 1.000; Z 1.000

I love my bubble

To create the “I love my bubble” we need a separated sphere. This allows us to animate it independent of the particle system. Hit [SHIFT+A], select UV Sphere and smooth it (bring up the Tool Shelf with [T] and hit Smooth) and name it ILoveMyBubble. Since this bubble will be right in focus we need to increase the number of faces to get a round silhouette. If we were to subdivide the sphere we would only subdivide the individual faces but the shape wouldn’t be any more round. If we apply the modifier SubSurf instead the whole shape will be recalculated and make it round for real. Another advantage is that we can keep the changes in the modifier stack and adjust it at anytime if we need to. Just as we did for the Main Camera and Main Light, head over to the Modifier window, open up Add Modifier and select Subdivision surface. The default values are fine for us.

Now we need a material with the text and the heart. I made a PNG in Photoshop CS4 but Photoshop does not save the alpha layer in a way that Blender likes so it didn’t work. I would recommend GIMP instead and make sure you untick all the PNG boxes when you save the image. You have to save it as a PNG to get the Alpha information included. Go to the Material window and with your new bubble selected hit the material list button and select our previously created Bubble Mat. Notice the little button named 2 beside the material name.

This indicates how many references this material has. We have 2 because the other reference comes from the particle system, using the same material. This also means that if we were to do any changes to this material, like adding a decal, it would change the look on all bubbles in the particle system as well.



ILoveMyBubble ▸ bubble Mat

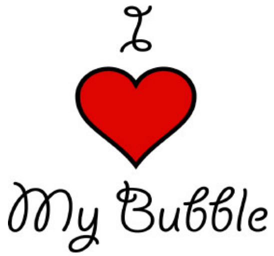
bubble Mat | 2 | Data

We need to put our decal in a new material slot so add a new slot by pressing the + button (beside the material slot list). A new slot is created and the previous selected material is now copied into a new slot.

Notice how the reference counter went up to 3 and this indicates that we really didn’t have a copy, but yet one more reference to the original material.

To make this material unique we need to unlink it by pressing the button with the number 3 on it. Now it's got a new name and the reference counter disappeared because there is only one object using this material now. Rename it to ILoveMyBubble Mat.

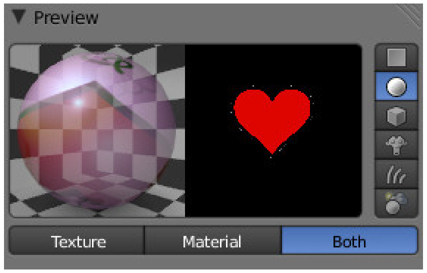
Go to the Texture window and select the next available texture slot and press the new button, rename it to ILoveMyBubble Img. Change the Type to Image or Movie. Go down to the Image panel and load the file ILoveMyBubble.png.



By Max Kielland

Tick the Anti-alias box to have the image anti-aliased when it stretches over the surface.

In the preview panel I usually set the option to see both the texture and what it will look like when used by the material. For some reason (bug?) the alpha is all black so you can't see the text. Don't be alarmed; as long it looks correct as a material its okay.

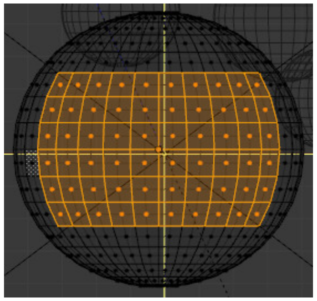


By default the image will be repeated in X and Y direction all over the surface. We only want this decal to appear once on the bubble so go down to the Image Mapping panel and change the Extension from Repeat to clip. The Extension defines what Blender should do when it reaches the edges of the image and in this case we just want it to stop, it will be just like an ordinary real world sticker.

If you have the sphere selected in the preview pane you will notice that our sticker is all distorted, but looks fine on a cube or plane. This is because we use the auto generated UV coordinates for our bubble. But changing this to sphere is not right either. The sticker gets distorted around the poles. In order to get the sticker applied right we need to create a new UV map to tell the sticker how it should be stretched over the surface.

First we need to create a UV Map slot. Go to the Object Data window and open up the UV Texture panel. Press the + button and name the slot ILoveMyBubble UV.

Wrap that bubble



Now we need to populate the UV map with coordinates and lucky us Blender has it all.

Head over to the UV editor with our Bubble selected. Make sure you look straight onto the bubble by pressing [Num1]. Enter edit mode with [TAB] and deselect all vertices with [A]. Hit [B] and draw a rectangle around the faces you want to show your label.

You can use any method you like to select the faces you want to display the texture.

When your surfaces are selected we are going to unwrap the area and create the UV coordinates. Press [U] to bring up the UV menu. It is important that you are in edit mode; otherwise you will get another menu. Select Project from view (bounds).

Now you will see your selected faces laid out over your texture. As you can see the mesh is not straight or the same size as the bubble's mesh. Don't be alarmed! The UV mesh is very different from the Object's mesh. The UV mesh has a “face” mapped to each selected face in the object's mesh, but the UV face size is not affecting the object's mesh size.

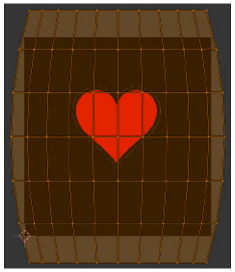
If two corresponding faces are of the same shape and size, the texture filling that face is undistorted. If the UV face is bigger it will cover more texture on the same physical face, resulting in a compressed texture. If it is smaller you get the opposite; a stretched texture instead.

By Max Kielland

There is a nifty function that shows how distorted the texture will be. To turn it on bring up the Properties panel [N] in the UV window. Under display tick the Stretch box.

If you select one vertex and move it around you can see how the colour changes depending on the amount of distortion to the texture. Turn off the Stretch box so we can work with the texture.

At the bottom you can select our previous loaded ILoveMyBubble.png image from the image list. Zoom out if you need to so we can see the whole texture.



The unwrap tries to fit the UV mesh inside the image but in this case we need the image to fit inside the UV mesh because our text is going all the way to the edges. Scale up the UV mesh with the [S] key. If you need to you can move the mesh to centre it over the image with [G].

That's it folks, now we have defined the coordinates for our UV map. Head back to the 3D view and go back to the Texture window. In the Mapping panel change Coordinates to UV. Select our ILoveMyBubble UV in the Layer field and use projection flat.

Now there is only one final step left before our texture is done. Head down to the Influence panel and tick the Color and Hardness boxes. Because the material is transparent our image will be very dim and we need to enhance it to be more visible. Change the Color to 6.000 instead of the default 1.000.

This panel tells Blender how the texture is interacting with the material itself. Color obviously will transfer the textures colours (in this case image) and Hardness will control the materials Hardness setting depending on the pictures colours.

Go back to the Material Window and select this new ILoveMyBubble Mat. As you can see we have 2 materials defined for this object, but how do we tell the object where to use this second material? You should still be in edit mode with your faces selected from the UV editor. This is perfect because it is precisely these faces we want to use our new material on. Under your material list (slots) press the Assign button to assign this material to the selected faces.

Move it, move it...

So far we have set up the whole scene, created all the materials, particle system and UV maps. Now it's time to make those bubbles float!

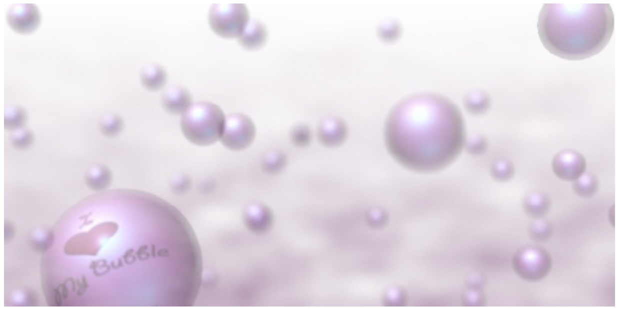
First I want to setup the animation format and size, so go to the Render windows Dimension panel. To speed up the process I used Resolution 640 x 360 with an Aspect Ratio of 1 x 1 at Frame Rate 25 fps.

Because we have all those transparent bubbles I want maximum Anti-Aliasing to smooth out all edges. Go down to the Anti-Aliasing panel and tick the box Anti-Aliasing and Full Sample. Set Anti-Aliasing to 16. The last step is to choose the output format. Go down to the Output panel and select the AVI Codec (or your favourite format). Select a folder where you want your AVI file.

Go to the Animation window.

The default setup of 250 frames will do fine for this tutorial; it will generate a 10 second animation. I want the I Love My Bubble to come into focus and show the decal and then disappear out again.

By Max Kielland



The easiest way is to start with the important thing first, positioning the bubble in focus.I want this to happen in the middle of the animation so I set the frame to 128 and move my bubble in position at -0.017, -13.824, 0.014 and rotation 0,0,0.

With the right frame selected and the bubble in position we will now insert a key frame with [I] and in the menu select LocRot. This will save the bubbles Location and Rotation values for this frame.

Now we only need to set a start and end position. Go to frame 1 and move the bubble to location -3.302, -15.991, -2.065 and rotation -90,-90,0. Insert a frame key with [I] and select LocRot again. Go to the last frame 250 and move the bubble to location 4.238, -8.606, -2.650 and rotation 90,90,0. Hit [I] again and select LocRot. Done!

The final picture for this article is at frame 99 but feel free to render the whole animation...

Where to go from here? Try for example different particle systems like the Boid, add collision deflection or play around with new particle shapes. Only your own imaginations are your limit...

I hope you have found this tutorial educational 🟥

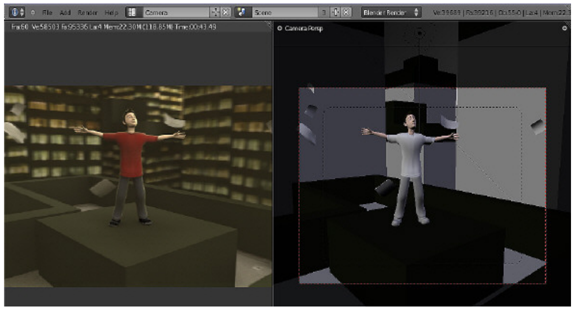
Up to Speed

By Nilson Juba & Jeff Israel

The new version of a program is always superior compared to the old one. But the new Blender 2.5 is really beyond what we expected.

We started a project before the launching of Blender's new version 2.5 and we felt the difference by manipulating the facilities between each version. The new vision is more attractive most at the interface. All the tools of the program are more quick and accessible. All these efforts made the new Blender 2.5 easier for the new users, of course.

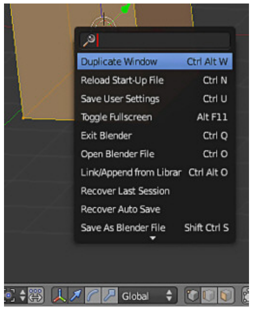
For those who are using Blender 2.49, the difference, for example in modeling , is that all the tools are more close to the hands.



The speed is incredible. Note that as we told before, we are working on project of an animation using the old version Blender 2.49 that we are finalizing

on the new version blender 2.5. In this work there is a scene where the camera makes a 360° round the top of the building with the main character at the center. All this work was mounted in Blender 2.49, but animated and rendered in Blender 2.5.

The time of rendering in Blender 2.49 was about 5 minutes. For rendering in Blender 2.5 the time was a breathtaking 35 seconds.


Duplicate Window Ctrl Alt W
Reload Start-Up File Ctrl N
Save User Settings Ctrl U
Toggle Fullscreen Alt F11
Exit Blender Ctrl Q
Open Blender File Ctrl O
Link/Append from Librar Ctrl Alt O
Recover Last Session
Recover Auto Save
Save As Blender File Shift Ctrl S
Global

The agility of this new version is also at the new features like the inclusion of a window with menu search that gives you a chance to find functions by name.

To activate this feature all you have to do is press the space bar on the 3D View 🟥

BISPODEJESUS BROS. Nilson Juba & Jeff Israel

We are two brothers from the southwest of Brazil, Nilson Juba and Jeff Israel.

Blog: http://bispodejesusbros.blogspot.com/

Twitter: http://twitter.com/bispodejesusb

Our contact: nilson.bispodejesus@gmail.com

It never takes the community long to produce needed documentation for new features in Blender. So it is no surprise that even though Blender 2.5 is still in beta, there are already a rather nice number of tutorials available to help you get the most out of the newest incarnation of Blender.

BLENDER COOKIE
MUNCH ON THESE

BlenderCookie.com has been using Blender 2.5 series for their video tutorials for some time now and has even started covering new features only found the latest test builds. In addition to full length video tutorials using 2.5, they have also been releasing shorter videos that cover specific tools and options in their Tips series. The BlenderCookie 2010 CG Cookie Blender Training Series is now live with Part 1 and the modeling section of part 2 available for purchase and download.

Blender Guru
Quality Blender Tutorials

BlenderGuru has been releasing tutorials done in 2.5, both in video and text format as well as new pdf ebook, The WOW Factor. He has started a new series of tutorials with each month focusing on a specific theme. I encourage you to check out the Weather series, his snow is amazing and his lightning tutorial is electrifying.

kernond
BlenderNewbies

Kernon Dillon (BlenderNewbies) has not only been producing video tutorials for 2.5, he is also busily working on a DVD full of brand new content for 2.5. And of course he has started a new series of modeling exercises that focus on technique and not just the end product, which are very educational and informative

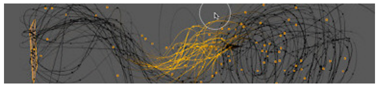
MAKING OF: Random Thoughts on Blender 2.5

By David Ward

In the three years that I've been working with Blender (since 2.43), I've seen about seven different upgrades to the software. Some of them were good (well, I guess ALL of them are good), but some I liked better than others.

For example, when the updated layout came out in, what, version 2.46 (?), I had a lot of difficulty letting go of the previous version, but eventually came to grips with it. As the new versions kept coming out, I came to look forward to these changes, especially when 2.48 was released along with the Big Buck Bunny project.

The new hair particle system blew me away, and I immediately jumped in and started playing with the hair and fur (even made me a nice a gorilla to test it out). The "Tree from Curves" script was also a handy feature, as it made Tree Creation relatively easy, if you knew what you were doing.

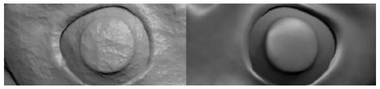


With the onset of 2.5, I still was hesitant to learn the new layout, as it was a huge change for me, and I'd never been one to edit the interface, or use the "dark" setting. But, since I was doing tutorials for others, it was conducive for me to upgrade my skills and learn the layout so I could help others to do the same. It has been a challenge doing tutorials for software that's still in development, but the interface has now become second nature to me, and I have no issues with it now (other than that it's still being developed and I'm antsy to see the final version).

The key features being brought in now will make Blender rival the "industry standards" even more so than it has in the past; previously, from what I could tell, the main lacking element was the

Volumetrics (y'know, clouds, fire, smoke, etc), but those now come standard, even in the beta version.

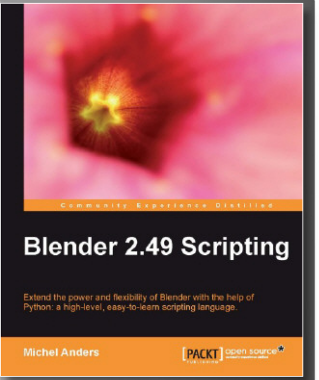
In addition, the Raytracing seems to be quite a bit faster; I noticed a few days ago when I was goofing around with it that the Raytraced Shadows rendered much faster than the Buffered Shadows, and that's not usually the case at all.



Maybe it was just the way I had the scene set up, but I think those developers know exactly what they're doing.

So with these new upgrades, plus the layout that will make Max and Maya users more comfortable, I'm really hoping to see more mainstream projects make use of Blender. I've been really impressed with what I've seen of Sintel, especially in the animation area, and am really excited to see the final product. Blender continues to amaze me, and the only thing I'd change about my experience with it is that I wish I would've started using it sooner.

It may be a few years out, but I can't imagine where we'll be at by version 3.0! Maybe then we'll have the "create awesome game" or "create awesome character" buttons by default 🟥


Community Experience Distilled
Blender 2.49 Scripting
Extend the power and flexibility of Blender with the help of Python: a high-level, easy-to-learn scripting language.
Michel Anders
PACKT open source

Review by - Satish Goda

"The wait for a comprehensive book on Python scripting in Blender 2.49 is over". I said this in my mind when I got to review the book titled "Blender 2.49 Scripting" by Michel Anders (and published by [PACKT] Publishing). I only wish this book had come out earlier, but I guess it's better late then never. What follows is my review of the book, arranged into sections.

Introduction

Blender is a premier open source tool for 3d content creation and its tool set and work flow has helped many an artist to realize his creativity into reality. Apart from its built-in tools, having the ability to extend the tool set and customize the software in a pipeline with Python was a bonus. The community has risen and developed quite a number of tools, ranging from simple to complex, using the Blender Python API.

But there wasn't a dearth of good tutorials on Python scripting in Blender and on using the API to create better animations and such. The books from Blender Foundation really helped bridge the gap between the software and novice users, but the missing link was a book for Python.

The wait is over. Enter "Blender 2.49 Scripting."

What does this book assume?

This book assumes that the reader is already familiar with using Blender and understands the data system. The basic concepts needed to script using Python are reviewed in each chapter. For example, the Object and DataBlock system, IPO and their uses, Poses etc., are reviewed so that one has good theoretical grounding before jumping into scripting.

The first chapter sets the groundwork by helping with installing Python. It also explains how Python scripts are automatically integrated into the menu and help systems using simple examples.

Learn By Example

One of the big strengths of this book is the breadth of programming/scripting examples across various aspects of Blender's toolset. From simple examples to intermediate and complex ones, the author lays down steps of the algorithm in simple English and then goes on to build the python code.

Especially commendable are the scripts/drivers for animating an IC engine. I have learned a lot of new techniques about using pydrivers and designing custom constraints.

The author makes use of Python library modules (internal and external) to create some very interesting scripts. For example, using Python's wave module to animate meshes using shape keys was a very good example of creating complex systems using existing tools.

Render management has also received good coverage. The examples for stitching images from various cameras was excellent, and using the Python Imaging Library is a very good example of system integration to get a job done well!

The best part for me was understanding how to extend Python's built-in script editor by writing plugins. On top of that, the examples on integrating the editor with SVN is simply amazing.

Support Files - Thank You!

Support files (blends, py scripts) provided with the book are indispensable and go hand in hand with reading the book.

Some of the chapters deal with writing complex tools, for which the author provides a nice walk through of the important pieces of the code. The full source code is provided with instructions on usage in Blender.

Also, the references chapter is a nice addition to the book. Links to Wikipedia pages are provided that cover the theoretical details of some of the example scripts. This was really helpful in illustrating the importance of research before implementing an idea.

What could have been Better?

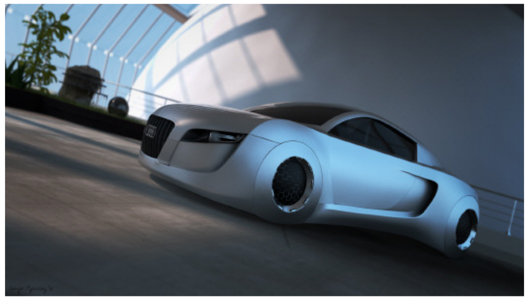
I believe that this book would have satisfied a much wider audience if there were very simple scripting examples at the beginning of every chapter. The chapter on programing custom mesh objects in Blender would have especially benefited.

Also, when the code for complex scripts is explained, the paragraphs could have been broken down for better readability.

More examples on adding OpenGL overlays to the 3D View would have been useful. I believe that the ability to do OpenGL programming in Blender is a really awesome feature and good examples on how to achieve this are few and far between.

In Summary

In summary, Blender 2.49 Scripting is a great technical and programming book for anyone interested in learning about the process of designing and implementing Python scripts for Blender ■



Rendering

by William Le Ferrand

Introduction

Corefarm.com and corefarm.org are rendering farms dedicated to high-resolution motionless scenes.

More and more Blender users have to render complicated scenes, but what a time-consuming task! Here we introduce not one but two rendering farms for Blender based on the Yafaray engine : corefarm.org, a community based grid where volunteers share their computers against credits, and corefarm.com, a professional solution for the most demanding users.

Have you been stuck by a heavy rendering? Then you know this torment: to have to wait for dozens or even hundreds of hours before getting the fruit of your labor - just to realize that a light is not set properly. You are not alone in this boat: architects, designers and 3d enthusiasts all over the world have faced those difficulties once, and everyone knows that rendering a 10000 x 10000 px poster is not an easy task.

What is a cluster?

A cluster is a group of connected computers dedicated to high performance computing.

Setting up a local infrastructure in your office to cope with your rendering needs is not an easy task: building a cluster (see frame) requires an disproportionate investment compared to the average needs of professional modelers. The standard way to overcome this barrier is to mutualize resources and this is precisely what renderfarms are about. Indeed, rendering farms are the easiest way to relieve your computers and to get your rendering done in impressive times. The modelling process is unchanged: you still use Blender locally and you still preview your work at low resolution locally.

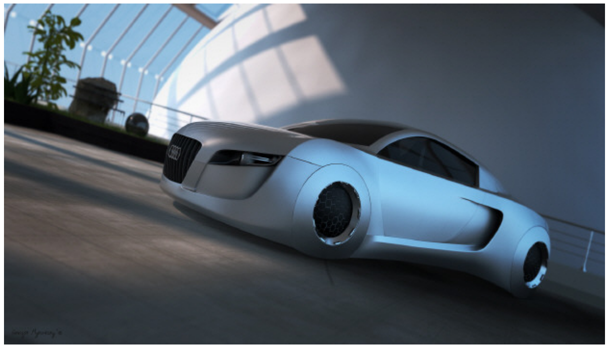
The novelty is that when you want to render your job at the final resolution, you simply upload your files to a remote server and start working on another project. The scene is split in the corefarm and each part of the scene is sent to a server; then the results are merged and you get a notification telling you that your image is ready - and you'll be surprised how fast you'll get it!

What is a Renderfarm?

A renderfarm is a cluster dedicated to computer-generated imagery. Although rendering farms were only set up for proprietary engines in past years, there is now a solution for Blender users based on the Yafaray engine: corefarm. Actually, there are two flavors of this farm, a collaborative edition and a professional one.

Corefarm.org is more an exchange place for CPU power than a standard farm: when your computer is idle, you share your computer with other 3d enthusiasts and when you need a burst of power to render your own scene, other users will share their computers with you!

by William Le Ferrand



A credit mechanism is here to monitor precisely who lends what amount of power to who. An important constraint with corefarm.org is that you have to participate to your own rendering - and to have a positive credit balance! Most of the corefarm.org code is open source and contributors help to continuously improve the service.

Corefarm.com is a standard rendering farm for Yafaray (v0.1.2): from the dedicated uploader you only have to select the XML file exported from Blender, you select precisely what amount of power you need: 25 GHz? 100 GHz? 500GHz?, and here it goes! Corefarm.com is also based on a credit mechanism: one credit gives you access to a 1GHz computer during one hour for your job.

How to render on corefarm.com?

1 Create an account on www.corefarm.com.

2 Download and install the corefarm uploader.

3 From Blender, using the Yafaray panel, export your work to a XML file.

4 From the corefarm uploader, select the XML file and the power you want to request, and upload.

5 We'll send you an email when the job is over!

Performance is here: impressive scenes are rendered daily on the corefarms. You want to give a try? Setting up a render is really easily (see frame) and you pay only for what you use. It is also pretty cheap! As modeled scenes get more and more complex, as rendering engines get more and more realistic, as resolutions increase, render farms will take more and more significance. They are the way for you, the 3d artists, to get your ideas sublimated into colorful pictures, as fast as imagination flies.

William Le Ferrand

We are two brothers from the southwest of Brazil, Nilson Juba and Jeff Israel.

Our contact: william@corefarm.com

You can get more information about the corefarms on

http://www.corefarm.org

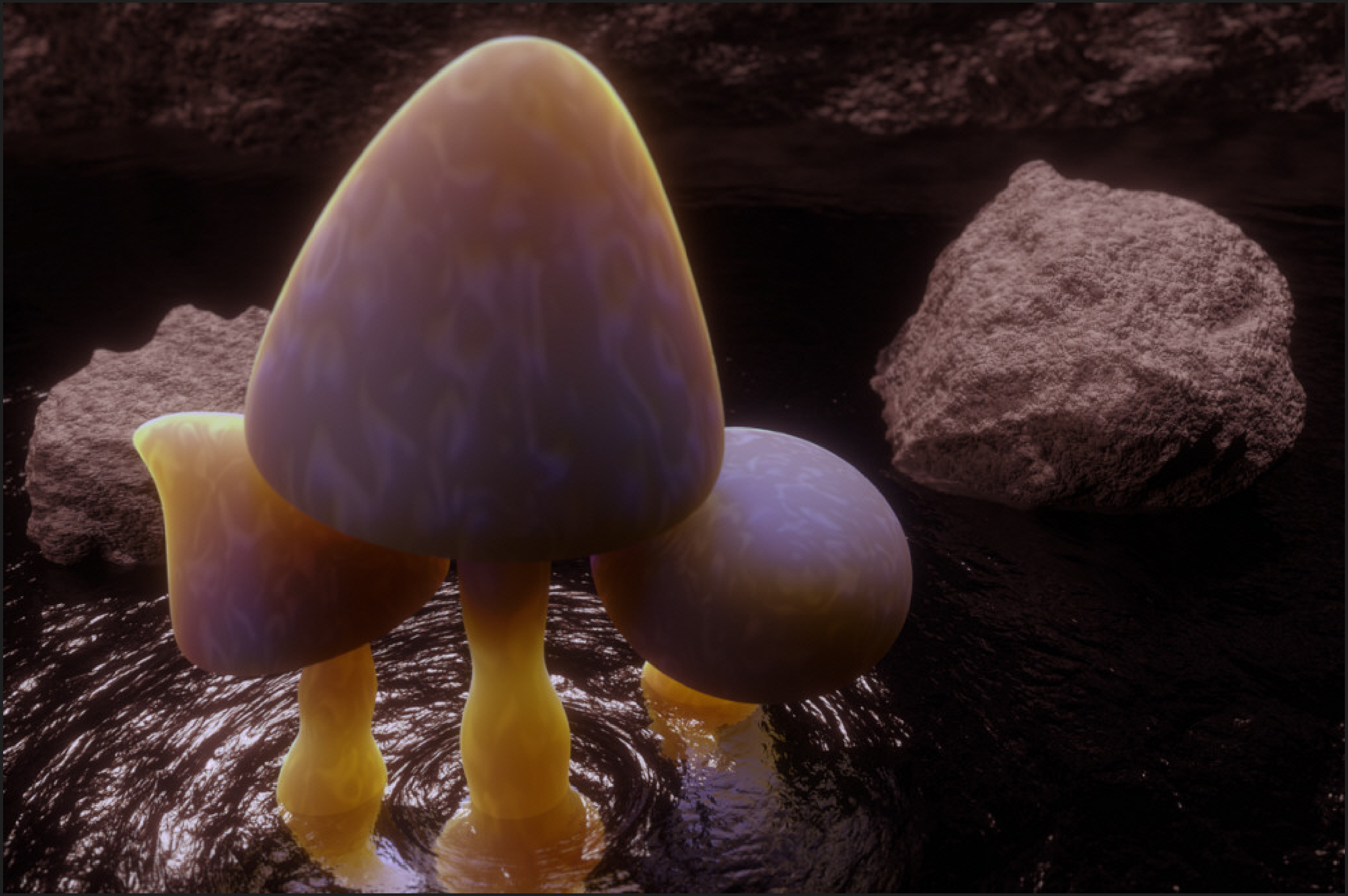
http://www.corefarm.com;

http://www.yafaray.org.

To keep updated, follow us on Twitter and/or Facebook. Happy rendering ■

Alone - by Will Davis

Dikarya - by Will Davis



KT - by Daniel D. Brown

Daniel DeWitt Brown 5/2010

Mobile - by David Jochems

Want to write for BlenderArt Magazine?

Here is how!

1. We accept the following:

- Tutorials explaining new Blender features, 3dconcepts, techniques or articles based on current theme of the magazine.
- Reports on useful Blender events throughout the world.
- Cartoons related to blender world.

2. Send submissions to sandra@blenderart.org. Send us a notification on what you want to write and we can follow up from there. (Some guidelines you must follow)

- Images are preferred in PNG but good quality JPG can also do. Images should be separate from the text document.
- Make sure that screenshots are clear and readable and the renders should be at least 800px, but not more than 1600px at maximum.
- Sequential naming of images like, image 001.png... etc.
- Text should be in either ODT, DOC, TXT or HTML.
- Archive them using 7zip or RAR or less preferably zip.

3. Please include the following in your email:

- Name: This can be your full name or blenderartist avtar.
- Photograph: As PNG and maximum width of 256Px. (Only if submitting the article for the first time)
- About yourself: Max 25 words .
- Website: (optional)

Note: All the approved submissions can be placed in the final issue or subsequent issue if deemed fit. All submissions will be cropped/modified if necessary. For more details see the blend-erart website.

Issue 29

"Industrial Revolution"

- Steam Punk
- Industrial Machines: big and small
- Factories and Industrial landscapes
- Pipes, Gears & Gadgets
- Grungy Materials suitable for industrial environments and objects

Disclaimer